UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

DESARROLLO DE UNA HERRAMIENTA PARA EL ESTUDIO DEL ESTADO DEL DNS EN DOMINIOS Y SU APLICACIÓN A .CL

JOSÉ ANDRÉS URZÚA REINOSO

Profesor Guía: Dr. Patricio Poblete

SANTIAGO DE CHILE
JULIO DEL 2003

Resumen

El buen funcionamiento y masificación que ha logrado Internet, se debe en gran parte a la existencia de los nombres de dominios. Por medio de estos, se puede acceder a distintos contenidos y servicios en cualquier lugar del mundo con sólo acordarse de un nombre, siendo esto posible gracias a la correcta operación del servicio de DNS. La existencia, importancia y funcionamiento del DNS generalmente pasa desapercibido para la mayoría de los usuarios mientras esté operando correctamente, y sólo se dan cuenta de su importancia cuando ocurre algún problema que no permite al usuario seguir accediendo a los contenidos que antes alcanzaba. Si el servicio de DNS dejase de funcionar, todas las comunicaciones que se realizan mediante Internet utilizando nombres de dominio fallarían (acceso a sitios web, envíos y recepción de emails, etcétera).

Esta tesis describe el desarrollo de una herramienta para el estudio del estado del DNS en dominios, la cual, luego de su ejecución, entrega un detallado reporte de los problemas encontrados, en que porcentaje de los nombres de dominios se presentan y una evaluación detallada para cada nombre de dominio y servidores DNS que se consultaron.

Para el desarrollo de la herramienta se estudiaron variados RFCs que describen los conceptos, funcionamiento y configuraciones del DNS, rescatando las características necesarias que debería cumplir un nombre de dominio para funcionar correctamente. Luego, se diseñó e implementó la herramienta, la cual alimenta una base de datos que mantiene todos los nombres de dominios, los servidores de nombres, las características evaluadas y las evaluaciones con sus glosas y resultados. A partir de esta base de datos, se genera un reporte con formato de página web, el cual muestra gráficos con los resultados obtenidos, permite consultar por una evaluación detallada por nombre de dominio y nombre de servidor y acceso a los recursos como código fuente y documentación.

El estado del DNS para .CL es regular, siendo la gran fuente de errores el descuido de los administradores y clientes de NIC Chile, los cuales no ponen especial cuidado en el momento de asociar los servidores de nombres al dominio. Por otro lado, se detectaron muchas configuraciones que no cumplen con las recomendaciones internacionales como nacionales, sobre temas de seguridad en los servidores DNS. Es necesario dar la importancia que merece al DNS, enseñarlo y fomentar las recomendaciones internacionales en el medio local, para que en un futuro no se lamenten los errores que se alertan hoy en día.

Índice general

Capítulo 1

Introducción

1.1. Antecedentes Generales

Cada día vemos que Internet es la base de nuevos servicios y negocios, sin importar mayormente como funcionan todos los procesos técnicos para que un servicio esté 'siempre disponible'. Es común que las personas estén acostumbradas a visitar contenidos en algún sitio web, simplemente ingresando la dirección de dicho sitio en un *browser* y luego esperar la respuesta que este programa le entrega, sin entender todo lo que sucedió detrás de la acción que acaba de ejecutar. De la misma manera, minuto a minuto, en diversos lugares del mundo se acceden a servicios en Internet siendo en un alto porcentaje de los casos exitoso el acceso.

La mayoría de los servicios disponibles en Internet son accedidos y recordados mediante un nombre, conocido como 'nombre de dominio', el cual es utilizado por el usuario para cuando desea acceder nuevamente. Para que estén siempre disponibles estos servicios, deben estar funcionando en uno o varios computadores en algún lugar del mundo, con el cual se establece una conexión cada vez que se accede. Los computadores en Internet son conocidos por un identificador único llamado *dirección IP*, utilizado para todo computador conectado a Internet. Entonces, la pregunta es: ¿Cómo las personas pueden acceder al servicio solicitado si no conocen la dirección IP del computador que presta ese servicio?. La respuesta es que existe algo llamado 'Domain Name System', que realiza un trabajo sumamente importante para el funcionamiento de Internet y permite que las

personas sólo recuerden nombres y el DNS se encarga de *traducir* ese nombre a la dirección IP del computador que presta el servicio solicitado.

La existencia, importancia y funcionamiento del DNS generalmente pasa desapercibido para la mayoría de los usuarios mientras esté operando correctamente, y sólo se dan cuenta de su importancia cuando ocurre algún error que no permite al usuario seguir accediendo a los contenidos que antes alcanzaba. Si el servicio de DNS dejase de funcionar, todas las comunicaciones que se realizan mediante Internet utilizando nombres de dominio fallarían (por ejemplo: sitios web, envíos y recepción de emails).

Dada la alta importancia del servicio de DNS para el funcionamiento de Internet, se han realizado estudios y análisis del funcionamiento de este servicio en los nombres de dominios, por parte de importantes entidades a nivel internacional como `RIPE` e `IETF`. Además, existen variados RFCs[1] que describen el funcionamiento estándar del DNS los que comúnmente no son respetados y pueden provocar un funcionamiento defectuoso de este servicio. Para el caso de los dominios .CL (y la mayoría de los `ccTLDs`), no existen estudios periódicos que muestren el estado del funcionamiento del DNS y menos si el estado es mejor o peor que hace un tiempo atrás.

Con esta tesis, se da paso a una investigación a fondo del servicio de DNS, revisando todas las recomendaciones aplicables para que este servicio funcione adecuadamente, un completo análisis a los estudios que se realizan en la actualidad y al diseño e implementación de una herramienta que permita determinar el estado del DNS en dominios, la cual se aplicará en un estudio particular para los dominios .CL. Como resultado de la aplicación de esta herramienta, se generarán informes periódicos mostrando el estado del DNS y los problemas encontrados con sus soluciones correspondientes, para así lograr conocer y mejorar el estado del DNS en los dominios.

1.2. Justificación

El servicio de DNS, su importancia y funcionamiento es bastante desconocido incluso en el ambiente de personas administradores de servidores de nombres, por lo que tener un estudio periódico que muestre el estado del DNS para los dominios que ellos sirven les sería de bastante utilidad, sobre todo para ir comparando como han evolucionado de hace un tiempo atrás.

[1] 'Request For Comments', documentos que definen los estándares de Internet.

Actualmente, no existe una herramienta gratuita que permita realizar un completo análisis del estado del DNS para algún nombre de dominio, pero sí existen herramientas comerciales las cuales pueden realizar análisis dependiendo del monto que se pague. Por otro lado, el crecimiento de Internet es cada día mas explosivo y va de la mano con la incorporación de nuevos nombres de dominios a la red, los cuales generan nuevos servidores de DNS que se incorporan a los ya existentes aumentando la probabilidad de que un error en algún servicio de DNS se propague más extensamente.

Para el caso de .CL, mientras el servicio de DNS esté funcionando (aunque sea momentáneo) es usual que los encargados de administrar los servidores de nombres no se preocupen de realizar estudios periódicos del estado del DNS, perdiendo la oportunidad de identificar a tiempo alguna pequeña falla que en el futuro podría generar un gran problema. Tampoco existe algún lugar o estudio general que les entregue información necesaria para los dominios.

Por otro lado, desde el mes de Junio del año 2003, existe el Consejo Nacional de Nombres de Dominios y Números IP, de la Subsecretaría de Telecomunicaciones del Gobierno de Chile, el cual, tiene como función el emitir recomendaciones a la comunidad sobre la gestión del sistema de nombres de dominio y los números IP en Chile. Dentro de las preocupaciones particulares de este consejo está el realizar estudios, encuestas y análisis de resultados en relación al funcionamiento de los nombres de dominios en Chile. Esto, sumado por el interés de NIC Chile, institución encargada de la administración del ccTLD .CL, en saber en que estado está el DNS para los dominios .CL, y agregando lo descrito en los párrafos anteriores, proveen la motivación para la investigación y trabajo desarrollado en esta tesis.

1.3. Objetivos

1.3.1. Generales

El objetivo general es realizar una evaluación de la calidad de los servicios de DNS en el país, a través de la generación de un reporte que muestre el estado actual del DNS para los dominios .CL.

1.3.2. Específicos

Los objetivos específicos que se deben lograr con esta tesis son:

- Investigar todos los estándares y recomendaciones aplicables para que el servicio de DNS funcione de manera adecuada.
- Analizar los estudios y herramientas que existen en la actualidad, preocupados de revisar el funcionamiento del DNS.
- Diseñar e implementar una herramienta que permita determinar el estado del DNS en dominios, generando un reporte con los problemas encontrados y soluciones correspondientes.
- Analizar el reporte obtenido, generando una publicación periódica del estado del DNS para los dominios estudiados.

1.4. Alcances

Esta tesis abarca sólo el estudio del servicio de DNS para los nombres de dominios, y no el estudio de otros servicios relacionados, como son el *web hosting*, *mail hosting*, etcétera.

La herramienta a desarrollar solo realiza el análisis para los dominios definidos en el primer nivel del nombre de dominio a analizar, por ejemplo para el caso de ejecutar la herramienta para .CL, el análisis se realizará sobre todos los subdominios de primer nivel de .CL, como: uchile.cl, nic.cl, pero no para: dcc.uchile.cl, www.dcc.uchile.cl, webmail.nic.cl.

Capítulo 2

DNS: Descripción General

2.1. Introducción

Luego de entender de manera básica la importancia del DNS, es necesario conocer un poco más a fondo su historia, los principales componentes, su funcionamiento y el desempeño actual que tiene este servicio, temas de los que trata este capítulo.

2.2. Historia

En la década de los años 70, la asociación entre los nombres de dominios y sus correspondientes direcciones IP era por medio de un archivo llamado **HOSTS.TXT**. Este archivo era mantenido por el *SRI*[1] *Network Information Center* (SRI-NIC), el cual se encargaba de transmitirlo por medio del protocolo FTP[2] a todos los hosts que participaban en Internet. Cada transmisión de un cambio en el archivo HOSTS.TXT, generaba un consumo de ancho de banda total proporcional al cuadrado del

[1] Stanford Research Institute
[2] File Transfer Protocol

número de hosts en la red. Por otro lado, al usar múltiples conexiones del servicio FTP se generaba una considerable carga en el servidor del SRI-NIC.

A medida que la red Internet crecía, aparecían nuevas organizaciones que deseaban administrar sus propios nombres de dominios y direcciones, pero debían esperar a que el SRI-NIC realizara los cambios pertinentes en el archivo HOSTS.TXT, ya sea para agregar un nuevo host con su dirección IP, para eliminar una asociación existente o para modificar. Este tiempo de espera cada vez era mayor, por lo que comenzaba a nacer el deseo de manejar el espacio de nombres de manera local. Por otro lado, no se permitían que dos hosts tuviesen el mismo nombre, lo que generaba exclusividad de nombres para quien primero lo utilizaba dentro del archivo HOSTS.TXT.

Las aplicaciones que funcionaban sobre Internet se fueron haciendo cada vez más sofisticadas, creando la necesidad de tener un servicio de nombres de dominios que mejorase todas las falencias de esquema que se estaba usando. Todas las propuestas que nacían eran diferentes, pero coincidían en que el nuevo esquema del funcionamiento del DNS debía ser de manera jerárquica. Así, la jerarquía en los nombres de dominio sería equivalente a las jerarquías existente en las organizaciones, usando el caracter punto (.) como separador entre jerarquías.

El responsable del diseño de la nueva arquitectura del DNS fue Paul Mockapetris, del USC´s Information Sciences Institute. En el año 1984 publicó los RFC 882 y 883, los cuales describen el **Domain Name System**, luego se publicaron los RFC 1034 y 1035, los cuales explican el funcionamiento actual del DNS. Estas publicaciones han sido complementadas por variados otros RFC que describen problemas potenciales de seguridad en el DNS, problemas de implementación, temas de administración, mecanismos de actualización dinámicos y más.

2.2.1. Metas del Diseño del DNS

Para el nuevo diseño del DNS, la primera meta era tener un espacio de nombres consistentes con el que será usado para referenciar los recursos. Así, se evitarían los problemas causados por nombres no adecuados a los recursos que direccionaban, ya que estaban compuestos por identificadores de red, direcciones IP, rutas o alguna información similar como parte del nombre.

La cantidad de hosts que se manejaban en Internet junto con la frecuencia de actualizaciones que sufrían, determinó que el nuevo sistema de nombres de dominios debería funcionar de manera

distribuida, con un registro de información local temporal (cache) para mejorar el funcionamiento. Las modificaciones y eliminaciones de registros de esa base de datos también debería ser de manera distribuida. Por otro lado, intentos por obtener una copia completa con la información consistente de la base de datos de nombres sería cada vez mas costosa y difícil de obtener, por lo cual, se debería evitar el intentar hacerlo. Estas ventajas y desventajas entre el costo de adquirir los datos, la velocidad de las actualizaciones y la precisión de los *cache* del nuevo esquema deben ser controlados por el encargado de proveer los datos a los demás hosts.

El nuevo diseño debería permitir que por medio del uso de los nombres se pudieran obtener direcciones de hosts, datos de casillas de email y alguna otra información que todavía no esté definida. Para esto, todos los datos asociados a un nombre serán de un tipo y las consultas que se generen para ese nombre podrían limitarse a un tipo.

El mismo espacio de nombres debería ser útil dentro de diferentes redes y aplicaciones, para lo cual el nuevo diseño del DNS debe permitir que funcione dentro de diferentes protocolos y administraciones. También se espera que las transacciones de un servidor de nombres sean independientes del sistema de comunicaciones que las hizo llegar al servidor. Algunos sistemas usarán datagramas para las consultas y respuestas, usando circuitos virtuales sólo para transacciones que requieran mucha confiabilidad (actualizaciones, transacciones muy largas), otros sistemas usarán siempre circuitos virtuales exclusivamente.

Finalmente, el sistema debería ser útil para una amplia variedad de capacidades de hosts, tanto los computadores personales como grandes servidores deberían ser capaces de usar el sistema.

2.2.2. Supuestos sobre el uso del DNS

Muchos de los problemas encontrados en los sistemas que usan bases de datos distribuidas son aplicables a este sistema de dominios, por lo que es necesario asumir ciertos supuestos sobre el uso que se le dará a este sistema. Estos supuestos son los siguientes:

- El tamaño total de la base de datos será inicialmente proporcional al número de hosts que están usando el sistema, pero eventualmente crecerá para ser proporcional al número de usuarios en estos hosts como a las casillas de correo electrónico y otra información que

se agregue al sistema de nombres de dominio.

- La mayoría de los datos en el sistema cambiará lentamente, pero el sistema debe ser capaz de actualizar esta información lo más rápido posible, en el orden de segundos o minutos.
- Cada organización que tiene bajo su responsabilidad un grupo de dominios deberá proveer servidores de nombres redundantes, que pueden estar dentro de la misma organización o fuera de ella.
- Los clientes del sistema de dominios deben ser capaces de identificar los servidores de nombres confiables, los que debe chequear antes de recibir referencias a servidores de nombres que no estén dentro de los confiables.
- El acceso a la información es más crítico que las actualizaciones instantáneas o garantías de consistencias. Cuando las actualizaciones no están disponibles, debido a problemas en la red o en algún host, el funcionamiento normal es mantener la antigua información mientras se sigue intentando actualizar. El modelo general establece que la información es distribuida con un tiempo asociado de refresco, el que distribuyó dicha información es responsable de asignar ese tiempo de refresco, y el que recibe la información es responsable de actualizarla una vez cumplido ese tiempo.
- Un servidor de nombres puede recibir una consulta que sólo puede ser respondida por otro servidor de nombres. Las dos formas de responder estas consultas son por medio de la **recursividad**, en la cual el primer servidor continua con la consulta recibida buscando una respuesta en lugar del cliente y la otra forma es la **iterativa** en la cual el primer servidor entrega la información que tiene en ese momento para la consulta y el cliente debe seguir consultando para obtener la respuesta. El sistema de dominios requiere la implementación de consultas iterativas, pero también permite las recursivas como opcionales.

Se supone que todos los datos originados en los archivos principales se esparcirán a través de los hosts que participan en el sistema de dominios. Estos archivos principales serán actualizados por los administradores del sistema local y serán archivos de texto que son leídos por un servidor de nombres local, para que luego estén disponibles a los usuarios del sistema de dominios por medio del servidor de nombres. Los programas de los usuarios accederán al servidor de nombres por medio de programas estándares llamados **resolvers**.

Los servidores de nombres y los resolvers serán configurados por un administrador de sistemas local. Para el caso del servidor de nombres, los datos de la configuración debe incluir identificadores de los archivos principales locales e instrucciones de cómo se debe obtener de servidores externos los archivos que no son principales para el servidor local. Para los resolvers, los datos de

configuración identifican el servidor de nombres que cumplirá la función de ser la principal fuente de información.

Para finalizar con los supuestos, los administradores de sistema deben proveer:

- La definición de los límites de las zonas
- Archivos principales de datos
- Actualizaciones a los archivos principales de datos
- Políticas de actualización de los datos

Por otro lado, el sistema de dominios entrega:

- Formatos estándares para los recursos
- Métodos estándares para consultar estos recursos
- Métodos estándares para que los servidores de nombres actualicen la información local desde servidores externos

2.3. Principales Elementos

El DNS tiene tres elementos principales: El espacio de nombres de dominio y los registros de recursos (Resource Records, RR en adelante), los servidores de nombres y los resolvers.

El espacio de nombres de dominios con los RR establecen las especificaciones para una estructura de tipo árbol, que mantienen los datos asociados a los nombres. De manera conceptual, cada nodo y hoja del árbol contiene un grupo de información asociada al espacio de nombres, y las consultas que se reciben extraen información de cierto tipo, de algún nodo u hoja del árbol.

Los servidores de nombres son programas que mantienen información sobre alguna parte del árbol de dominios y algunos punteros a otros servidores de nombres que pueden ser usados para obtener la información de cualquier otra parte del árbol de dominios. Los servidores de nombres deben saber para que parte del árbol de dominios tienen una completa información, para los que se dice que es un servidor de nombres **AUTORITATIVO** para esa parte del árbol de dominios. La información autoritativa es organizada dentro de elementos llamados *zonas*, las que pueden ser distribuidas de manera automática a otros servidores por motivos de redundancia.

Los resolvers son programas que extraen información desde un servidor de nombres, en respuesta a algún requerimiento de un cliente. Además, deben ser capaces de acceder hasta el último servidor de nombres y usar la información que este le entrega ya sea para responder la consulta al cliente o para seguir buscando las respuestas, utilizando las referencias que recibió. Los resolvers típicamente son accedidos directamente desde los programas de los usuarios, no necesitando un protocolo especial para esta comunicación.

Desde el punto de vista del usuario, el sistema de dominios es accedido por medio de procedimientos que realizan llamadas al resolver local. El espacio de dominios consiste en un árbol, y el usuario puede solicitar información de cualquier parte de ese árbol. Para el resolver, el sistema de dominios está compuesto por un desconocido número de servidores de nombres, y cada servidor de nombres tiene una o más piezas del árbol de datos, pero el resolver visualiza cada una de estas *bases de datos* como estáticas. Para el caso del servidor de nombres, el sistema de dominios consiste en grupos separados de información llamados *zonas*, para las cuales tiene copias locales y que debe actualizar periódicamente. Además, recibe consultas desde los resolvers solicitando información de alguna zona.

2.3.1. Espacio de Nombres y Registros de Recursos

Espacio de Nombres

El espacio de nombres tiene forma de árbol, en donde cada nodo y hoja corresponde a un grupo de información, pero que también puede estar vacía. Cada nodo tiene una etiqueta, la cual va de 0 a 63 caracteres de largo y los nodos hermanos no pueden tener la misma etiqueta, pero si puede usarse la misma etiqueta para nodos que no son hermanos. Existe una etiqueta reservada y es la etiqueta vacía (de largo cero) la cual es usada para la raíz.

El nombre de dominio de un nodo es la lista de etiquetas desde el nodo hasta la raíz del árbol. Por convención, las etiquetas que componen un nombre de dominio son leídas desde la izquierda a la derecha, desde la más específica (la más lejana a la raíz) a la menos específica (terminando con la raíz). Para crear los nombres de dominios, se tiene la libertad de usar la misma etiqueta con minúscula o mayúscula pero no ambas para hermanos, es decir, es la misma etiqueta si se escribe con mayúsculas o minúsculas.

Cuando un usuario desea escribir un nombre de dominio, separa las etiquetas con puntos (.), como un nombre de dominio completo termina con la etiqueta de la raíz (nombre vacío), este se visualizará con un punto al final.

También se debe distinguir entre los nombres de dominios *absolutos* y *relativos*. Los absolutos son nombres de dominio completos, por ejemplo: *paola.ejemplo.cl.*, los nombres de dominios relativos muestran solo la etiqueta inicial del nombre, la cual se completa por medio de los programas locales que tiene conocimiento del dominio local, para el caso anterior un dominio relativo seria: *paola*, usado en el dominio *ejemplo.cl*. Para simplificar las implementaciones, la suma total de los caracteres que representan un nombre de dominio está limitada a 255.

Un dominio es subdominio de otro dominio si está contenido en el nombre absoluto del dominio, por ejemplo A.B.C.D. es un subdominio de B.C.D., C.D., D. y '.'.

Antes de que el DNS sea usado para manejar toda la información del espacio de los nombres de dominios, es necesario conocer dos reglas básicas:

- Una convención para traducir desde un objeto nombre a nombres de dominios. Esto describe como la información sobre un objeto es accedida.
- Tipos de RR y formatos de datos para describir el objeto.

Al aplicar estas reglas para los hosts, la traducción depende de la sintaxis existente para los nombres de hosts la cual es un subconjunto de la representación textual de un nombre de dominio, asociado con los formatos del RR para describir la dirección del host. Debido a la necesidad de tener un sistema confiable de traducción inversa desde una dirección de host a un nombre de host, se definió una traducción especial en el dominio IN-ADDR.ARPA.

Para el caso de las casillas de email, la traducción es un poco más compleja que para los hosts. Una dirección como: nombre-local@dominio-mail, es traducida en un nombre de dominio convirtiendo nombre-local en una etiqueta (obviando los puntos que este contenga) y convirtiendo dominio-mail en en un nombre de dominio utilizando el formato de texto común para los nombres de dominios (los puntos actúan como separadores de dominios), y concatenando los dos resultados en un nombre de dominio. Esto haría que la casilla de email: paola@ejemplo.cl sea representada por el nombre de dominio: paola.ejemplo.cl.

El usuario típico no está enterado de la definición de estas reglas, pero debería entender que ellas son el resultado de numerosos compromisos entre las decisiones que se toman para mantener compatibilidad con las antiguas funcionalidades, interacciones entre objetos diferentes y el inevitable impulso de agregar nuevas funcionalidades cuando definieron estas reglas.

Un diagrama ejemplo que representa el espacio de nombres de dominios es el de la figura 2.1. En donde se aprecia la raíz del árbol denotada por el caracter punto (.), y luego los dominios de primer nivel: com, edu, gov y mil, los cuales pueden tener más subdominios. En este caso, el dominio de primer nivel edu, tiene como subdominios a *isi.edu.* y *mit.edu.*. Finalmente, el dominio isi.edu, tiene como subdominios a: *vaxa.isi.edu.* y *venera.isi.edu.*.

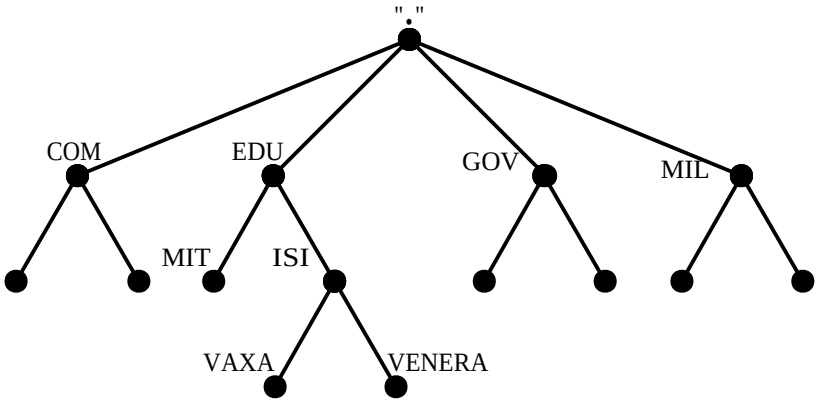

Figura 2.1: Ejemplo del Espacio de Nombres de Dominios.

El espacio de nombres de dominio, no impone alguna regla particular en las etiquetas de un nombre de dominio, y tampoco agrega algún significado especial a las etiquetas de un nivel en particular. Cuando se administra una parte del espacio de nombres de dominios, los administradores deciden cual será la semántica y significado de los nombres que usarán. Sin embargo, el espacio de nombres de dominio existente tiene una estructura auto impuesta. Especialmente en los dominios de más alto nivel, estos nombres siguen ciertas tradiciones las que son necesario conocerlas y

entenderlas para tratar de descifrar correctamente un nombre.

La organización original de los nombres de dominios de más alto nivel dividió el espacio de nombres en 7 dominios:

1. *COM*: Destinado para las organizaciones comerciales, como: hp.com, sun.com, ibm.com.
2. *EDU*: Destinado para las organizaciones educacionales, por ejemplo: berkeley.edu, purdue.edu.
3. *GOV*: Destinado para las organizaciones gubernamentales, como nasa.gov.
4. *MIL*: Destinado para las organizaciones militares, como: army.mil, navy.mil.
5. *NET*: Formalmente, esta destinado para las organizaciones que proveen infraestructura de redes, como nsf.net, uu.net. Sin embargo, desde 1996, NET como COM no han tenido restricciones de uso.
6. *ORG*: Destinado para organizaciones no comerciales, como eff.org. Pero tal como sucedió con NET, desde 1996 las restricciones sobre el dominio ORG han sido eliminadas.
7. *INT*: Destinado para organizaciones internacionales readas por tratado, como nato.int.

Actualmente, estos nombre de dominios originales son conocidos como *Generic Top-Level Domain*, o *gTLDs*. Cerca del año 2001, se agregaron siete nuevos dominios a estos, los que incluían a *BIZ*, *NAME*, *INFO* y *PRO* con el fin de acomodar la rápida expansión de Internet y la necesidad de más nombres del espacio de dominios. La organización responsable de la administración del DNS de Internet es la `ICANN`.

Con el fin de adecuar el incremento de la internacionalización de Internet a mediados de los años 80, a los dominios de primer nivel que describían organizaciones, se agregaron otros que describían áreas geográficas. Asi, nuevos nombres de dominios de primer nivel fueron reservados para corresponder a países individuales, usando abreviaciones de dos letras para cada país en el mundo, los que son llamados *Country Code Top Level Domain*, o *ccTLDs*. Esta lista de dominios de primer nivel y de los nombres de los países se puede apreciar en la sección 8.2 del capítulo 8.

Registro de Recursos (RRs)

Un nombre de dominio identifica a un nodo en el árbol del espacio de nombres de dominios, este nodo tiene un conjunto de información para ese dominio, la cual puede ser vacía. El conjunto de información asociada a ese nombre de dominio en particular, está compuesta por registros de recursos (RRs), el orden de los RRs no es importante y no necesita ser preservado por los servidores de nombres, resolvers u otra parte del DNS.

Los RR establecidos en el RFC 1034[3], contienen la siguiente información:

- **Dueño**(owner): Indica cual es el nombre de dominio que contiene este RR.
- **Tipo**(tipo): Especifica el tipo de recurso, es un valor codificado de 16 bits. Los tipos se refieren a recursos abstractos, los que son:

A	Una dirección de host
CNAME	Especifica el nombre canónico de un alias
HINFO	Muestra infomación del host, como el Sistema Operativo y CPU
MX	Especifica un servidor de emails para el dominio.
NS	Indica un servidor de nombres autoritativo para el dominio
PTR	Es un puntero a otra parte del espacio de nombres de dominios
SOA	Identifica el inicio de una zona de autoridad

- **Clase**(class): Indica la familia del protocolo en el cual es válido este RR, se usan las siguientes clases:

IN	El sistema Internet
CH	El sistema Chaos

La clase Chaos es usada para especificar datos en la zona para la red CHAOSnet del MIT, un protocolo creado a mediados de los años 70. Actualmente, está clase es utilizada por algunos proveedores de software DNS, para mantener cierta información. Por ejemplo, el

software Bind utiliza esta clase para mantener la versión del software, entregando esta información cuando el servidor recibe una consulta de tipo txt con clase chaos por el registro 'version.bind'[3]

- **Tiempo de Vida**(TTL): Indica el tiempo de vida del RR, es un entero de 32 bits que está en unidades de segundos. Es usado principalmente por los resolvers, cuando almacenan RR en su memoria cache, en donde el TTL les indica cuanto tiempo lo deberían mantener en su memoria antes de borrarlo.
- **Datos del Recurso**(RDATA): Corresponde a los datos que describen el recurso y depende del tipo de recurso:

Tipo	**RDATA**
A	Para la clase Internet es una dirección IP de 32 bits. Para la clase CHAOS es un nombre dominio seguido por una dirección octal chaos de 16 bit
CNAME	Un nombre de dominio
MX	Un valor que indica la preferencia del servidor de emails (el menor tiene mayor preferencia), seguido por el nombre de host, el cual actuará como servidor de correos para el dominio
NS	Un nombre de host
PTR	Un nombre de dominio
SOA	Varios valores, el primero es el nombre del servidor primario, luego el email de la persona a cargo de esta información. A continuación, vienen los tiempos de refresco, reintento, tiempo de vida y tiempo de expiración de la información, los cuales son explicados más adelante.

El tiempo indicado por el TTL, no es aplicable para datos autoritativos en las zonas, los que tienen un tiempo de expiración que está definido en las políticas de actualización de la zona. El TTL es asignado por el administrador de la zona, en donde se originan los datos. Mientras un TTL pequeño minimiza el uso de cache para la información, un TTL con valor cero lo impide. Para un buen funcionamiento de Internet se sugiere un tiempo de TTL de un par de días, pero si se tiene prevista una modificación a la información con anticipación, este tiempo se podría disminuir, y cuando el cambio esté realizado y la información actualizada se vuelve al tiempo original del TTL.

Dentro del protocolo del DNS, los RRs son representados en forma binaria, y se representan de

[3]Utilizando la herramienta dig, la consulta deberia ser: 'dig @nameserver version.bind txt chaos', en donde 'nameserver' es el nombre del servidor a consultar.

una manera altamente codificada cuando se almacenan en un servidor de nombres o en un resolver. En el siguiente ejemplo se mostrará definiciones de RRs, los cuales, al comienzo de la línea tienen el dueño del RR. Si la línea comienza con un espacio en blanco, se supone que el dueño de ese RR es el mismo que el del RR previo. A continuación del dueño, se muestra el TTL, la clase y el tipo del RR. Finalmente, se agrega la sección RDATA, en donde se ingresa la información necesaria de la clase del RR.

```
urzua.cl.        86400  IN  NS ns.vulcano.cl.
                 86400  IN  NS ns.urzua.cl.
                 86400  IN  MX 10 mail.urzua.cl.
ns.urzua.cl.     86400  IN  A 192.80.24.30
                 86400  IN  A 200.54.68.140
mail.urzua.cl.   86400  IN  A 192.80.24.45
```

En el ejemplo anterior, se comienza definiendo 3 RRs para el dueño *urzua.cl.*, estos RRs tienen como TTL 86400 segundos y son todos del tipo Internet (IN). Los dos primeros corresponden al tipo NS, indicando que los servidores de nombres del dominio *urzua.cl.* son *ns.vulcano.cl.* y *ns.urzua.cl.*. El último RR definido para este dueño es del tipo de servidor de emails (MX), en donde la prioridad de atención es 10 (sería útil si existiese otro RR de esta misma tipo para este mismo dueño) y el nombre del host que actuará como servidor de emails para este dominio es *mail.urzua.cl.*. Luego se definen 2 RR para el dueño *ns.urzua.cl.*, los cuales tienen el mismo TTL y tipo que los anteriores. Estos dos RRs son de la tipo **A**, que definen una dirección IPv4 para el dueño. En este caso, se definen 2 direcciones IP para el host *ns.urzua.cl.*, las que son: *192.80.24.30* y *200.54.68.140*. Finalmente, se define un último RR para el dueño *mail.urzua.cl.*, de tipo A, definiendo la dirección IP indicada para este nombre de host.

En los sistemas actuales, los hosts y otros servicios son obtenidos de nombres que identifican el mismo recurso. Por ejemplo, los nombre *mail.urzua.cl.* y *www.urzua.cl.* pueden identificar el mismo host. De la misma manera, muchas organizaciones proveen varios nombres que actúan con el mismo servidor de emails. Por ejemplo *jose@a.urzua.cl.*, *jose@b.urzua.cl.* y *jose@urzua.cl* pueden ser la misma casilla de email, aunque el mecanismo que exista detrás de esto puede ser muy complicado.

Muchos de estos sistemas tienen la noción de que existe un nombre canónico o primario y el resto sólo son alias a ese nombre primario. El DNS provee esta características por medio del uso de los RR de tipo CNAME (canonical name). Un RR de tipo CNAME identifica a su dueño como un

alias a un nombre primario definido en la sección RDATA del registro. Si un CNAME está presente en un nodo del árbol de dominios, no debe estar acompañado de ningún otro tipo de información, para así asegurar que la información del nombre canónico y su alias es la misma.

Los registros de tipo CNAME generan acciones especiales en los softwares del DNS. Cuando un servidor de nombres falla al buscar un RR solicitado en el conjunto de recursos asociado al nombre de dominio, este revisará si el conjunto de recursos consiste en un registro de tipo CNAME. De ser así, el servidor de nombres incluirá el registro CNAME en la respuesta y reiniciará la consulta de el nombre de dominio especificado en el campo de datos del registro CNAME.

Por ejemplo, un servidor de nombres está procesando una consulta por el nombre *www.urzua.cl.*, preguntando por información del tipo *A* y obtiene los siguientes RRs:

```
www.urzua.cl.     IN     CNAME     web.urzua.cl.
web.urzua.cl.     IN     A         10.0.0.57
```

Los dos RRs serán retornados en la respuesta a la consulta que solicita información del tipo *A*, y se retornará solo la información de tipo CNAME para cualquier otro tipo de consulta por el registro (de tipo CNAME u otro). Los nombres de dominio que apuntan a otro nombre siempre deberían apuntar al nombre primario y no a otro alias. Esto generaría referencias extras y ciclos bastante largos de terminar en la búsqueda de la información de un nombre de dominio. El RR que indica la dirección IP del host con nombre primario deberia ser:

```
57.0.0.10.in-addr.arpa.    IN      PTR     web.urzua.cl.
```

Si este registro PTR tuviese en su sección RDATA al nombre *www.urzua.cl.* se generaría un ciclo, que debería ser indicado como un error.

2.3.2. Servidores de Nombres y Zonas

Servidores de Nombres

Los servidores de nombres son repositorios de información que activan la base de datos de uno o varios dominios. La base de datos es dividida en zonas, las cuales se distribuyen a lo largo de los servidores de nombres. La tarea esencial de un servidor de nombres es responder las consultas que recibe, usando la información que posee en las zonas. Por diseño, los servidores de nombres pueden responder las consultas de manera simple, en la cual se genera la respuesta sólo con la información local, entregando en la respuesta la información deseada o una referencia a otro servidor en donde obtenerla.

Un servidor de nombres típicamente mantiene una o más zonas, pero tendrá autoridad sólo sobre una pequeña parte de la información del árbol del espacio de nombres de dominios. Además, mantendrá en su memoria cache información de otros dominios para los cuales no tiene autoridad. El servidor de nombres indicará en sus respuestas si tiene autoridad para la información que está entregando, de está manera el que realiza la consulta puede saber si la respuesta proviene de una fuente de información con autoridad o no.

Las consultas que llegan al servidor de nombres y las respuestas que entrega este, son transportadas en un mensaje con formato estándar descrito en el RFC 1035[4]. Las consultas contienen en QTYPE, QCLASS y QNAME, las cuales describen el tipo, clase de la información deseada y el nombre que les interesa.

Los servidores de nombres pueden actuar de manera recursiva o no-recursiva, la forma más simple de operar es la no-recursiva, donde el servidor puede responder la consulta usando sólo la información local que posee, y la respuesta puede tener un error, la respuesta o una referencia a otro servidor. Todos los servidores de nombres deben implementar este tipo de respuesta no-recursiva. La manera simple para un cliente es que el servidor responda de manera recursiva. En este modo, el servidor de nombre actúa como un resolver, y retorna un error o una respuesta, pero nunca una referencia. Este servicio es opcional para los servidores de nombres, y el servidor que implemente este tipo de respuesta debe tomar control sobre los clientes que usan este modo recursivo.

El uso del modo recursivo es limitado a los casos en que el cliente y el servidor de nombres están

de acuerdo en este uso. El acuerdo es negociado a través de dos bits en los mensajes de consulta y respuesta. Un servidor de nombres indicará la disponibilidad de respuestas recursivas indicando en todas sus respuestas el bit RA (Recursion Available) como encendido. El bit RA indicará disponibilidad de recursión. Para el caso de las consultas, esta contendrá el bit RD (Recursion Desired), por medio de cual el cliente indicará si requiere el servicio de recursividad para esta consulta. Los clientes esperarán respuestas recursivas de los servidores que previamente les han enviado el bit RA, o de los cuales tienen acuerdos previos obtenidos fuera del protocolo del DNS.

Si el servicio de respuestas recursivas está disponible y lo requiere un cliente, la respuesta recursiva a la consulta será una de las siguientes:

- La respuesta a la consulta, con los RRs encontrados durante la búsqueda de la respuesta. También puede ser uno o más CNAME encontrados.
- Un error de nombre, indicando que el nombre no existe.
- Indicación de un error temporal.

Cuando no se tiene disponible el servicio de respuestas recursivas o no se requiere por parte del cliente, la respuesta no-recursiva será una de las siguientes:

- Un error, generado con autoridad, indicando que el nombre no existe.
- Una indicación de que un error temporal ocurrió.
- Alguna combinación de:
  - Un grupo de RRs con la respuesta a la pregunta, acompañados con una indicación de si la respuesta se obtuvo de las zonas que tiene el servidor de nombres o de la información que mantiene en cache.
  - Una referencia a un servidor de nombres el cual tiene las zonas con la información del nombre requerido.
- Un grupo de RRs que el servidor de nombres cree que serán útiles al cliente.

Un ejemplo de resolución recursiva de una consulta es el de la figura 2.2, en donde un resolver envia una consulta DNS para obtener la dirección IP del host *jose.urzua.cl.*.

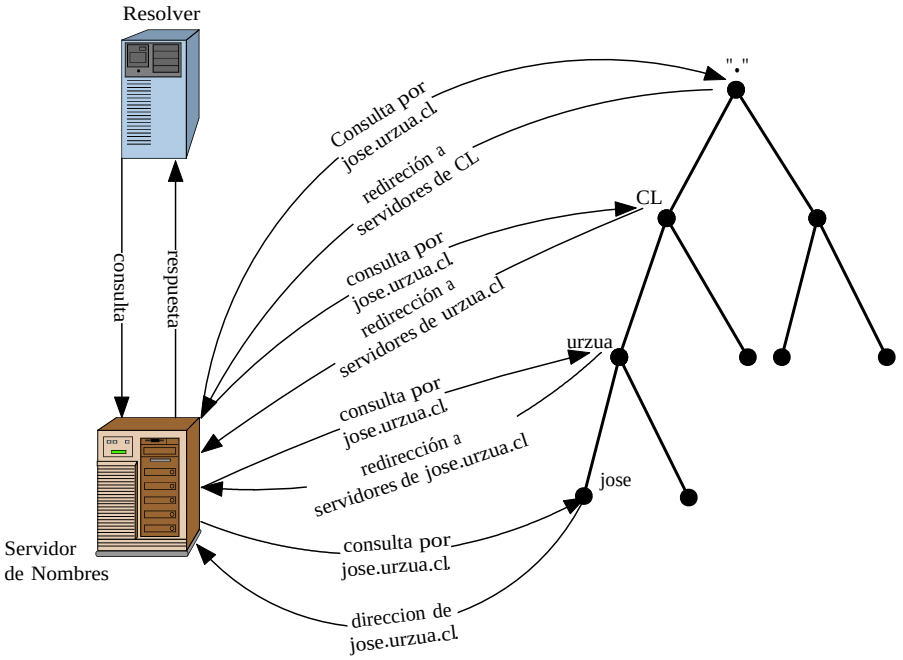

Figura 2.2: Ejemplo de resolución con recursividad.

Zonas

Una zona es una parte del espacio de nombres de dominios, la cual se obtiene desde un archivo o de algún servidor de nombres, y debería estar en varios servidores de nombres, para así asegurar su disponibilidad en caso de alguna falla de enlace o de algún host. Los datos que están presentes en una zona estan organizados en 4 partes principales:

1. Datos autoritativos para todos los nodos que están dentro de la zona.
2. Datos que definen el nodo principal de la zona.
3. Datos que describen sub-zonas delegadas.
4. Datos que permiten el acceso a las sub-zonas, también conocidos como datos *glue*.

Todos estos datos están expresados en forma de RRs, y también una zona puede ser completamente descrita en términos de un grupo de RRs. Mientras las zonas pueden ser transferidas entre

los servidores de nombres mediante transferencias de RRs, otros las llevan mediante una serie de mensajes o por medio del protocolo FTP con un archivo principal que lleva una representación textual de la zona.

Los datos autoritativos de la zona es toda la información de los RRs anexados a todos los nodos desde el nodo principal hasta los nodos hojas o los nodos que marcan el límite de la zona. Los RRs que describen el nodo principal es especialmente importante para la administración de la zona. Estos RRs son de dos tipos: RRs de la clase de servidores de nombres en una lista para todos los servidores de la zona o un único RR de tipo SOA que describe los parámetros de administración de la zona.

Los registros que describen las sub-zonas son los de tipo NS, que son los servidores de nombres para estas sub-zonas. Desde que los límites de la zona es entre nodos del árbol del espacio de nombres de dominios, estos RRs no son parte de la información autoritativa de la zona, y debería ser exactamente los mismos RRs definidos en el nodo principal de esta sub-zona.

Una de las metas de la estructura de las zonas, es que cualquier zona tenga todos los datos necesarios para establecer comunicaciones con el servidor de nombres de cualquier sub-zona. De esta manera, la zona padre tiene toda la información necesaria sobre los servidores que permiten acceder a sus zonas hijas. Si el nombre de un servidor de nombre de una zona hija está formado por parte del nombre de dominio del nodo padre (como un sub-dominio), dentro de la información de la zona se debe agregar un registro *glue* de tipo *A*, el cual indicará la dirección IP asociada a ese servidor de nombres. Esta información será utilizada para las respuestas que llevan referencias.

A continuación se muestra una zona ejemplo para el nombre de dominio *urzua.cl.*:

```
@  IN  SOA  ns.urzua.cl.  jose.urzua.cl. (
                      2003093013     ; version(yyyymmddhh)
                      7200           ; refresh
                      3600           ; retry
                      604800         ; expire
                      86400)         ; TTL
urzua.cl.      IN    NS     ns.urzua.cl.
urzua.cl.      IN    NS     ns.vulcano.cl.
urzua.cl.      IN    MX     10 mail.urzua.cl.
urzua.cl.      IN    A      192.80.24.40
```

```
ns.urzua.cl.    IN    A      200.2.212.146
                IN    A      200.1.19.1
mail.urzua.cl.  IN    A      192.80.24.23
www.urzua.cl.   IN    CNAME  mail.urzua.cl.
```

Esta zona, comienza definiendo el registro de tipo SOA para para el dominio *urzua.cl.*, el cual tiene como primer parámetro el nombre del servidor de nombres primario para este dominio: *ns.urzua.cl.* y luego viene la dirección de correo electrónico del responsable de esta zona, que se traduce a: *jose@urzua.cl*. Los parámetros siguientes del registro SOA son la versión de la zona: *2003093013* que está formada con el número del año concatenado con el número de mes y día, anexando al final un correlativo de dos números que podría corresponder a la hora de la última modificación dentro de ese día. A continuación vienen los tiempos de refresco, reintento, expiración y TTL de la zona.

El tiempo de refresco (7200 segundos) indica la frecuencia con la que un servidor secundario consultará si ha ocurrido un cambio en la información de la zona al servidor primario, el tiempo de reintento (3600 segundos) será con la frecuencia que un servidor secundario intentará realizar el refresco una vez que el refresco normal falle, el tiempo de expiración (604800 segundos) indicará por cuanto tiempo la información que contiene la zona es válida antes de descartarla cuando el servidor primario no es accesible y el tiempo de TTL (86400) indicará por cuanto tiempo la información de la zona puede ser almacenada en cache.

Luego se definen dos registros de tipo NS, que actuarán como servidores de nombres del dominio *urzua.cl.*, los que son: *ns.urzua.cl.* y *ns.vulcano.cl.*. A continuación se define un registro de tipo MX que indica que el servidor de correos es *mail.urzua.cl.* con prioridad 10. Finalmente, se definen todos los registros de tipo *A* para el dominio, en particular el host *ns.urzua.cl.* tendrá dos direcciones IP, las que son: *200.2.212.146* y *200.1.19.1* y el registro *www.urzua.cl.* es un alias al nombre primario *mail.urzua.cl.*.

Se debe hacer notar que dentro de la zona sólo se definen los registros de tipo A para los subdominio del dominio padre (*urzua.cl.*), para el caso del registro NS *ns.vulcano.cl.* no se agrega un registro de tipo A, indicando la dirección IP de este host, pues la dirección IP debe estar definida en la zona del dominio *vulcano.cl.*.

Una de las metas del diseño del DNS es la descentralización de la información, esto se logra mediante la *delegación* en las zonas. La delegación en los dominios es similar a la delegación de

tareas en un trabajo. Un administrador debe separar un gran trabajo en pequeñas tareas y delegar la responsabilidad de cada una de esas tareas a diferentes empleados. En la figura 2.3 se puede apreciar un ejemplo de delegación de nombres de dominios.

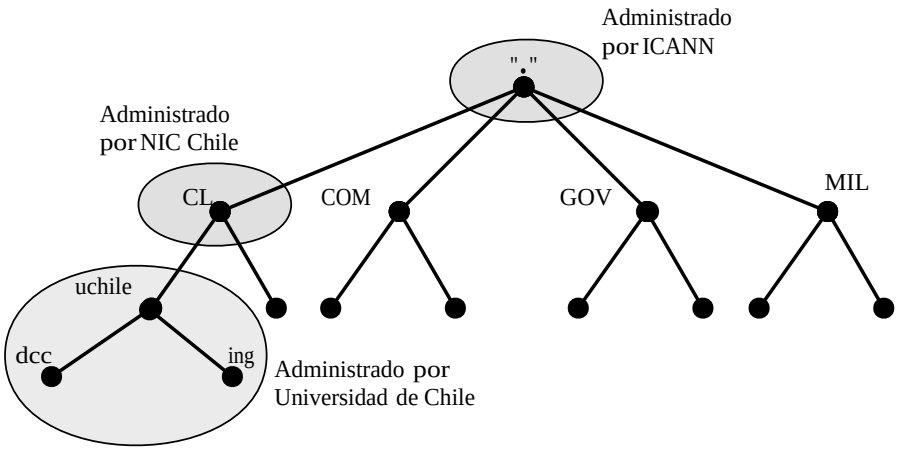

Figura 2.3: Ejemplo de delegación de nombres de dominios.

Una organización que maneja un nombre de dominio puede dividirlo en subdominios. Cada uno de estos subdominios puede ser *delegado* a otras organizaciones. Esto significa que una organización llega a ser responsable de mantener todos los datos en estos subdominios, teniendo la libertad de cambiar los datos y de la misma manera dividir este subdominio en más subdominios y delegarlos. El nombre de dominio padre, mantiene sólo punteros hacia las fuentes de datos de sus subdominios delegados para dirigir los requerimientos hacia ellos.

2.3.3. Resolvers

Los resolvers son programas que funcionan como una interfaz entre los programas de usuario y los servidores de nombres de dominio. En un caso simple, un resolver recibe requerimientos desde un programa de un usuario (programas de envío de emails, Telnet, FTP) en forma de una llamada de una sub-rutina o llamada de sistema, y responde con la información solicitada en una forma compatible con el formato de los datos del host local.

Un resolver podría estar ubicado físicamente en la misma máquina que los programas que solicitan sus servicios, pero el resolver necesitará consultar servidores de nombres que están funcionando en otros hosts. El tiempo que el resolver usará para responder una consulta, puede variar

desde milisegundos a varios segundos, esto se debe a que el resolver puede obtener las respuestas consultando a servidores externos u obtener la información desde la memoria cache local.

Un objetivo importante del resolver es eliminar el tiempo de espera de respuesta de las redes y la carga en los servidores de nombres por una alta cantidad de requerimientos. Este objetivo lo cumple respondiendo los requerimientos usando su memoria cache, como primera prioridad. Esto permite que los cache que son compartidos por múltiples procesos, usuarios y máquinas son más eficientes que los cache no-compartidos.

La interfaz típica de un cliente con un resolver tiene tres funciones principales:

1. **Traducción de un nombre de host a una dirección IP.** Esta función es a menudo definida como para imitar las funciones que cumplía el archivo HOSTS.TXT. Dado un grupo de caracteres, el cliente esperará una o más direcciones IP de 32 bits. Bajo el sistema del DNS esto se traduce en requerimientos que solicitan RRs de tipo *A*. Dado que el DNS no conserva el orden de los RRs, esta función que consulta podría tomarse el tiempo de ordenar las direcciones que obtuvo o seleccionar la mejor si el programa cliente sólo recibe una dirección IP como respuesta. Se recomienda que en las respuestas se retorne todas las direcciones IP que se encontraron.

2. **Traducción desde una dirección IP a un nombre de host.** Dada una dirección IP de 32 bits, el cliente espera recibir un grupo de caracteres que identificarán un nombre de hosts. Los octetos de la dirección IP serán invertidos, usados como componentes de un nombre, y se agregará el sufijo *IN-ADDR.ARPA*. Una consulta de tipo PTR será usada para obtener el RR con el nombre primario del host. Por ejemplo, un requerimiento por el nombre de hosts correspondiente a la dirección IP *1.2.3.4* buscará los RRs de tipo PTR para el nombre de dominio *4.3.2.1.IN-ADDR.ARPA*.

3. **Funciones generales de búsqueda.** Esta función obtiene información arbitraria desde el DNS, y tiene contraparte en otros sistemas. El cliente provee un QNAME, QTYPE y QCLASS y espera recibir todos los RRs que concuerdan con el requerimiento. Esta función a menudo usa el formato del DNS para todos los datos que necesita.

Cuando el resolver decide que función es la que utilizará, comúnmente obtendrá uno de los siguientes resultados antes de pasarlos al cliente:

- Uno o más RRs que entregan los datos requeridos. En este caso, el resolver entrega la respuesta en el formato adecuado.
- Un error de nombre (Name Error, NE). Esto sucede cuando el nombre solicitado no existe. Por ejemplo, cuando el usuario comete algún error al escribir el nombre.
- Un error de datos no encontrados. Esto sucede cuando el nombre que se solicita existe, pero los datos del tipo apropiado para la respuesta no se encuentra. Por ejemplo, al aplicar la función de buscar la dirección IP de un host a una casilla de email, aunque el nombre exista se generará el error por que el RR de tipo *A* no existe.

Es importante hacer notar que las funciones de traducción entre un nombre de host y una dirección IP, combinarán los errores de nombre y los de datos no encontrados, retornando un único tipo de error, pero en general esta función no debería hacerlo. Una razón para esto, es que la aplicación preguntará primero por un tipo de información del nombre, seguido por un nuevo requerimiento al mismo nombre pero por alguna información distinta, si los dos errores se combinan, las consultas inservibles harán mas lenta la aplicación.

Mientras un resolver se presta a responder un requerimiento particular, se puede encontrar con que el nombre en cuestión es un alias. Por ejemplo, el resolver está usando el nombre dado (que es un alias) para traducir del nombre de host a una dirección IP, por lo que encuentra un RR de tipo CNAME. Si es posible, el resolver retornará al cliente el alias que obtuvo, dependiendo de las condiciones. En muchos casos, el resolver simplemente reinicia la consulta con el nuevo nombre cuando encuentra un CNAME. Sin embargo, esto no debería realizarse por parte del resolver, pues pueden existir consultas en donde el QTYPE es por un CNAME, en donde el usuario está interesado en saber el RR de tipo CNAME por sí mismo, y no el RR que es apuntado por el alias.

Variadas condiciones especiales pueden ocurrir con el uso de alias. Múltiples niveles de alias deberían afectar la eficiencia con la que se obtienen las respuestas, pero no debería indicarse como un error. Ciclos de alias y alias que apuntan a nombres que no existen deben generar un error, el cual debe ser entregado al cliente.

En la figura 2.4, se puede apreciar un esquema de funcionamiento general entre un resolver, un servidor de nombres y otros resolver y servidores externos.

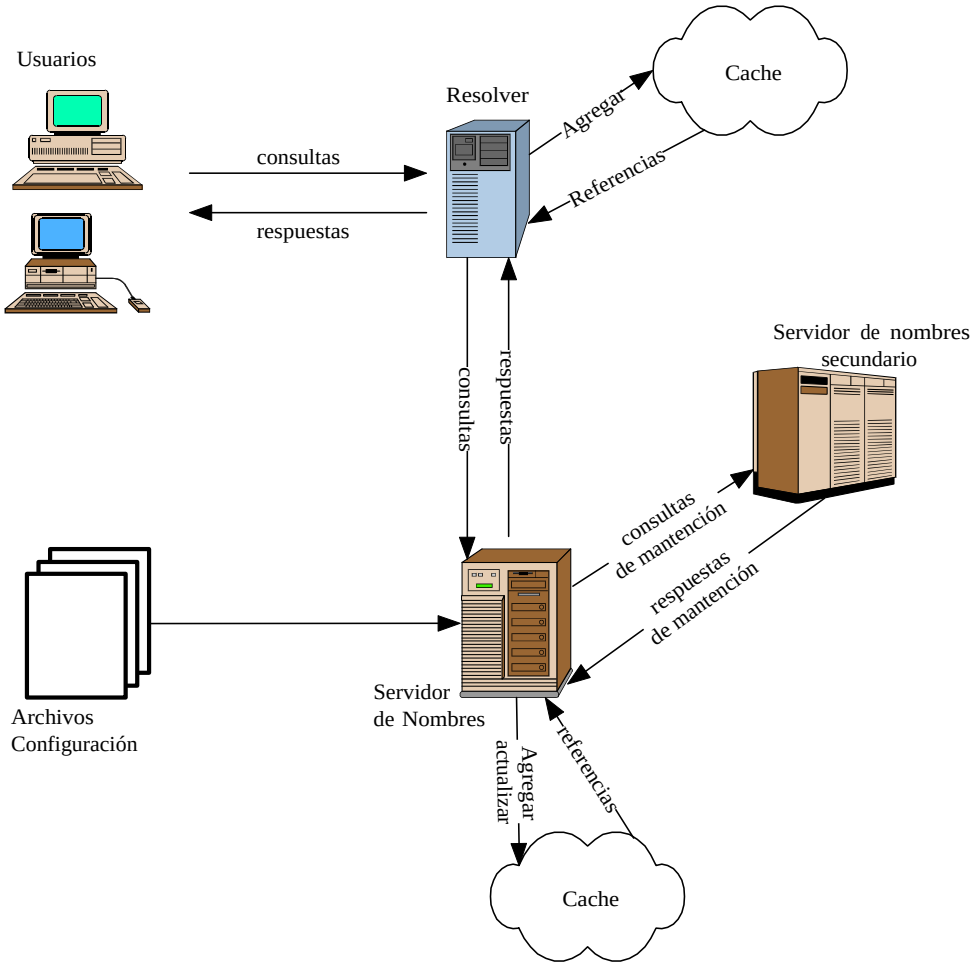

Figura 2.4: Ejemplo de Funcionamiento General.

2.4. Funcionamiento

Una vez que se conocen los principales elementos que forman el DNS, es necesario saber como interactúan entre ellos. Los servidores de nombres reciben mensajes para los cuales deben generar una respuesta, estos mensajes se conocen como *queries*. En Internet las queries son transportadas en datagramas *UDP*. La respuesta del servidor de nombres tendrá la respuesta a la pregunta, referencias a otro grupos de servidores, o alguna señal de error.

2.4.1. Consultas (Queries)

En general, el usuario no genera consultas directamente, pero al solicitar los servicios de un resolver este generará y enviará una o más consultas a los servidores de nombres y manejará las condiciones de error y las referencias que puede recibir como respuestas.

Las consultas (queries) DNS y sus respuestas son transportadas en mensajes con un formato estándar. El formato del mensaje tiene un *header* que contiene un número fijo de campos, los cuales siempre deben estar presentes, y cuatro secciones que contienen los parámetros de la consulta y los RRs.

El campo más importante en el header es un campo de 4 bits llamado opcode, el cual separa diferentes consultas. De los 16 valores posibles, uno es parte del protocolo oficial (consulta estándar), dos son opciones (consulta inversa y consulta de estado), uno es obsoleto y el resto no están asignados todavía. Las otras cuatro secciones son:

- **Question**: Contiene el nombre de la consulta y otros parámetros de ella.
- **Answer**: Contiene los RRs que responden directamente la consulta.
- **Authority**: Contiene los RRs que describen los servidores con autoridad. Opcionalmente, contiene el RR de tipo SOA para los datos que recibió con autoridad en la sección de *Answer*.
- **Additional**: Contiene los RRs que serán de utilidad al usar los RRs de las otras secciones.

Se debe hacer notar que lo descrito es el contenido, no el formato, el cual varía con el header opcode.

2.4.2. Algoritmo de un Resolver

Cuando un resolver recibe un requerimiento de un usuario, generará una o más consultas DNS hacia los servidores de nombres adecuados. El algoritmo que utiliza el resolver para buscar las

respuestas a las consultas, utiliza ciertas estructuras de datos que representan el estado de un requerimiento, las que son:

- SNAME: El nombre del dominio para el cual se realizará la búsqueda.
- STYPE: El QTYPE de la búsqueda solicitada.
- SCLASS: Corresponde al QCLASS de la búsqueda solicitada.
- SLIST: Es una estructura que describe los servidores de nombres y la zona en la cual el resolver está actualmente tramitando la consulta. Esta estructura mantiene la información de cual es el mejor servidor de nombres para la información que se desea buscar. Esta estructura contiene el equivalente a una zona, con el servidor de nombres de la zona, direcciones conocidas de ese servidor de nombres, e información histórica que puede ser usada para sugerir algún servidor de nombres como el mejor para responder alguna consulta.
- SBELT: (Safety Belt) Es una estructura del mismo tipo que SLIST, la cual es inicializada desde un archivo de configuración, y una lista de servidores que deberían ser usados cuando el resolver no encuentra en la información local alguna guía para seleccionar un servidor de nombres adecuado.
- CACHE: Estructura que almacena los resultados de las búsquedas ya realizadas. Desde que los resolvers son los responsables de eliminar los RRs para los cuales ha expirado su TTL, muchas implementaciones convierten ese TTL en un tiempo de ordenamiento cuando el RR es almacenado en el cache.

El algoritmo de más alto nivel utilizado por un resolver tiene cuatro pasos, que son:

1. Revisar si la respuesta está en la información local, si es así, retornarla al cliente.
2. Buscar los mejores servidores para consultar.
3. Enviar las consultas a los mejores servidores hasta que alguno retorne una respuesta.
4. Analizar la respuesta:
    - Si la respuesta responde la pregunta o contiene un *name error*, se almacena la información en cache y se responde al cliente.

- Si la respuesta contiene una delegación a otro servidor, se almacena en cache la delegación y vuelve al paso 2.
- Si la respuesta contiene un CNAME y no es lo solicitado, se almacena en cache el CNAME, se cambia el SNAME al *canonical name* en el CNAME RR y vuelve al paso 1.
- Si la respuesta muestra una falla en el servidor u otro contenido erróneo, elimina el servidor de SLIST y vuelve al paso 3.

En el paso 1 se busca en cache por la información deseada. Si la información está en cache, se asume que es la mejor para el uso normal. Algunos resolvers tienen una opción en la interfaz de usuario que permite forzar al resolver a ignorar la información de cache y consultar a un servidor con autoridad. Esto no es recomendado por defecto.

En el paso 2 se busca un servidor de nombres para preguntar la información requerida. La estrategia general es buscar en los RRs disponibles localmente, comenzando con SNAME, luego el dominio padre de SNAME, luego el padre del padre de SNAME, y así sucesivamente hasta llegar hasta la raíz.

En el paso 3, se envían las consultas hasta que una respuesta es recibida. La estrategia es hacer un ciclo con todas las direcciones de todos los servidores con un tiempo de espera entre cada transmisión. En la práctica, es recomendable usar todas las direcciones IP de un servidor. La estructura SLIST contiene típicamente los valores para controlar los tiempos de espera y mantener alguna señal de las transmisiones previas.

En el paso 4 se analiza la respuesta. El resolver debería revisar que la respuesta corresponde a la consulta enviada por medio del campo ID en la respuesta. La respuesta ideal es de un servidor con autoridad para la consulta el cual entrega la información solicitada o un error de nombre. Los datos son enviados al usuario y se almacena en cache para un uso futuro si el TTL es mayor que cero. Más detalles de implementación se pueden revisar en el RFC 1035 [4].

2.4.3. Algoritmo de Resolución

Los algoritmos actuales utilizados por los servidores de nombres dependen directamente del sistema operativo y de las estructuras de datos utilizadas para almacenar los RRs. El algoritmo que se describirá asume que los RRs son almacenados en una estructura con forma de árbol, uno por cada zona, y otros para el cache:

1. Asignar o sacar el valor que indica disponibilidad de recursión en la respuesta, dependiendo si el servidor de nombres provee este servicio o no. Si el servicio de recursividad está disponible y el requerimiento lo solicita (por medio del bit RD), se debe ir al paso 5, de lo contrario al paso 2.

2. Buscar en las zonas disponibles por la zona ancestro más cercana a QNAME. Si una zona es encontrada se va al paso 3, de lo contrario al paso 4.

3. Comienza la búsqueda hacia abajo, etiqueta por etiqueta, en la zona. El proceso de búsqueda puede tomar diferentes caminos:

   - Si el QNAME es encontrado, se ha encontrado el nodo. Si la información del nodo es un CNAME, y el QTYPE no es un CNAME, copia el RR de tipo CNAME en la sección de respuesta, cambia el QNAME por el nombre canónico en el RR de tipo CNAME, y vuelve al paso 1. Si no, copia todos los RR que coinciden con QTYPE en la sección de respuesta y va al paso 6.
   - Si algún calce con la respuesta está fuera de la zona autoritativa, se tiene una referencia. Esto sucede cuando se encuentra un nodo con RRs de tipo NS determinando el límite al final de la zona. Se copia los RRs de tipo NS para la sub-zona en la sección de autoridad de la respuesta. Se ponen las direcciones disponibles en la sección adicional, usando *glue RRs* si la dirección no está disponible en los datos autoritativos de cache. Se va al paso 4.
   - Si para alguna etiqueta no se logra un calce, es decir, la etiqueta correspondiente no existe, se busca si la etiqueta * existe. Si la etiqueta * no existe, se revisa si el nombre que se está buscando es el QNAME original en la consulta o es un nombre que se tiene debido a un CNAME. Si el nombre es original, se asigna un error de nombre de manera autoritativa en la respuesta y retorna. Otro caso también retorna.

     Si la etiqueta * existe, compara este RRs contra el QTYPE. Si se encuentra un calce, lo copia en la sección respuesta, pero asigna el dueño del RR como el que viene en QNAME, y no el nodo con la etiqueta *. Se va al paso 6.

4. Comienza la búsqueda en el cache. Si QNAME es encontrado en el cache, se copia todos los RRs anexados a él, que calzan con QTYPE, en la sección respuesta. Si no existe delegación para los datos autoritativos, se busca el mejor desde el cache, y se asigna en la sección de autoridad. Se va al paso 6.

5. Usar un resolver local o una copia del algoritmo del resolver para responder la consulta. Almacenar el resultado, incluyendo cualquier CNAME intermediario, en la sección respuesta.

6. Usando solo información local, se agregan otros RRs que pueden ser útiles para la sección adicional de la consulta. Terminar.

En el algoritmo descrito, se puede apreciar que se entrega un tratamiento especial a los RRs, cuyo nombre de dueño comienza con *. Estos RRs son llamados *wildcards*, y son usadas como instrucciones para sintetizar RRs. Cuando se dan las condiciones apropiadas, el servidor de nombre crea RRs con un nombre de dueño igual al nombre que viene en la consulta, y el contenido se toma desde el RRs de tipo wildcard.

Los contenidos de un RR de tipo wildcard sigue las reglas y formato estándar de los RRs. Los wildcard en la zona tienen un dueño que controlará las consultas por nombres que ellos calzarán. El nombre del dueño del wildcard es de la forma: *.dominio, en donde dominio es cualquier nombre de dominio y no debe contener otra etiqueta * dentro de él, y debería ser un dato autoritativo para la zona. Los wildcard son aplicados para los descendientes del dominio, pero no para el dominio mismo.

Para mostrar el uso de los RRs de tipo wildcard, suponga el nombre de dominio de una institución llamado: *universidad.cl.* y un servidor llamado *mail.universidad.cl.* el cual actúa como servidor de emails para el dominio. Además, los siguientes registros están en la zona del dominio *universidad.cl.*:

```
universidad.cl.           IN    MX     10 mail.universidad.cl.
*.universidad.cl.         IN    MX     10 mail.universidad.cl.

mail.universidad.cl.      IN    A      146.83.4.11
mail.universidad.cl.      IN    MX     10 mail.universidad.cl.

*.mail.universidad.cl.    IN    MX     10 mail.universidad.cl.
```

Estos registros presentes en la zona, podrían causar que cualquier requirimiento por un registro de tipo MX de cualquier nombre de dominio que termine en *universidad.cl.* retornará un registro que apuntará a *mail.universidad.cl.*. El efecto del primer wildcard (**.universidad.cl.*) es eliminado al colocar información explicita para el registro *mail.universidad.cl.* (registro A y MX respectivamente), por lo que para mantener el efecto del wildcard es necesario agregarlo nuevamente al final.

Capítulo 3

Problemas Comunes en el DNS

3.1. Introducción

El DNS por ser un sistema distribuido, puede verse afectado por una variedad de problemas cada vez que se agrega un nuevo servidor de nombres a la red, el análisis de estos problemas, clasificados en problemas de configuración y operación son presentados en este capítulo.

3.2. Problemas de Configuración

Los problemas de configuración en el DNS tiene como origen los datos que las personas tienen en sus servidores de nombres, encontrados típicamente en los archivos de zona que el servidor de nombres mantiene en memoria durante su operación.

3.2.1. Datos Inconsistentes, Faltantes o mal Formados

Todos los host que integran Internet deberían tener un nombre, y los servicios disponibles en Internet no podrán comunicarse con dicho host si no está correctamente registrado en el DNS. También se debe tener la seguridad que los RR de tipo A y PTR coinciden, definiendo los PTR en el dominio *in-addr.arpa*. Si el host tiene mas de una dirección IP se debe asegurar que todas las direcciones IP tiene su correspondiente registro PTR. Además, los registros PTR deben apuntar a un registro de tipo A válido y no a un alias definido por medio de un CNAME.

Los nombres de dominios consisten en etiquetas separadas por puntos y el DNS entrega bastantes libertades para definir un nombre en un dominio. Sin embargo, si un nombre de dominio es usado para nombrar a un host, este debería respetar las restricciones que existen para los nombres de host[5], y lo mismo sucede cuando se quiere utilizar como dirección de emails[6]. Se debe tener especial cuidado con los nombres que pueden tener distintos significados dependiendo de la sintaxis, por ejemplo *0xe* es un nombre válido, pero si se solicita el servicio: *telnet 0xe* este podría tratar de conectarse a la dirección IP *0.0.0.14*.

Algunos sistemas operativos tienen algunas limitaciones en el largo de los nombres de host, mientras esto no sea un problema del DNS se deben tomar las precauciones necesarias en el sistema operativo local antes de asignar un nombre de host.

Cabe mencionar que muchos RR tienen uno o más de un argumento, por ejemplo los RR HINFO y RP. Si no se proporcionan los argumentos necesarios, los servidores podrían retornar datos con contenido impredecible para los argumentos que faltan.

3.2.2. Registros SOA

Para todas las zonas, en el registro SOA se debe agregar la dirección de email de la persona encargada de mantener el DNS para el dominio en cuestión. El símbolo ‘@‘ en el email debe ser reemplazado por el ‘.‘, y no se debe tratar de poner el ‘@‘ en esta dirección. Si la parte del nombre de usuario de la dirección de email contiene el caracter ‘.‘, por ejemplo: Jose.Urzua@ejemplo.cl, se debe anteponer el caracter ‘\‘, quedando el ejemplo como: Jose\.Urzua.ejemplo.cl. Se debe tener especial cuidado con los software que generan de manera automática los valores del registro SOA,

los cuales pueden colocar direcciones de email con formatos erróneos. Es muy importante que esta dirección de email sea una dirección válida, ya que a menudo es usada para reportar datos erróneos en el DNS o algún incidente de seguridad.

Para el número serial de la zona se recomienda una sintaxis del estilo: YYYYMMDDnn[2], en donde YYYY es el año, MM corresponde al número del mes, DD al número del día y nn a un número que identifique la versión de la zona de ese día (podría ser la hora).

Se deben asignar valores lógicos para los tiempos del registro SOA, el tiempo de refresco (refresh) debería estar de acuerdo al tiempo que se desea que los servidores secundarios mantengan los datos sin realizar una actualización con el servidor primario. Se puede utilizar un tiempo pequeño (20 minutos a 2 horas) si no se está preocupado por el consumo del ancho de banda, o un tiempo grande (2 a 12 horas) si la conexión a Internet es lenta. El tiempo de reintento (retry), no es tan importante como los otros, a menos que los servidores secundarios estén en alguna red alejada del servidor primario. Típicamente es una fracción del tiempo de refresco. El tiempo de expiración (expire), debe ser tan grande como el tiempo que uno desee que la información se mantenga en los servidores secundarios cuando estos no se logran comunicar con el servidor primario. Después de este tiempo, el servidor secundario seguirá tratando de comunicarse con el servidor primario, pero no prestará el servicio de DNS para la zona. Se recomienda un valor entre 2 a 4 semanas. El tiempo de TTL (time to live) corresponde al tiempo que se espera que los RR estén presentes en la memoria cache de los servidores de nombres, y debe estar de acuerdo a la variación de los datos de la zona, mientras mayor sea este tiempo más se demorarán las propagaciones de las actualizaciones a los datos de la zona, un TTL con valor cero prohibiría el almacenamiento en cache, haciendo más lenta la respuesta al requerimiento. Los valores típicos están entre 1 a 5 días.

3.2.3. Registros Glue

Los registros *glue* son registros de tipo A que están asociados con registros de tipo NS para entregar una completa información al servidor de nombres. Por ejemplo:

```
urzua.cl.        86401  IN  NS   ns.urzua.cl.
                 86401  IN  NS   ns2.urzua.cl.
ns.urzua.cl.     86401  IN  A    192.80.24.41
ns2.urzua.cl.    86401  IN  A    146.83.4.11
```

En este caso, los registros de tipo A son llamados registros *glue*, pues definen las direcciones IP de los servidores de nombres del dominio, ya el que nombre de estos servidores, depende del nombre de dominio padre (urzua.cl.).

Si un servidor de nombres tiene varias direcciones IP, se deben agregar todas estas direcciones en la zona de los registros glue, para así evitar inconsistencias en los cache, que causarán que algunas búsquedas no encuentren a todas las direcciones de los servidores de nombres.

Existe una mala costumbre de agregar registros glue cada vez que se agrega un registro NS, para asegurarse de que el registro NS tenga asociada una dirección IP. Esto generaría registros duplicados en la zona, lo que hace más difícil la actualización de dirección de ese registro (por tener que identificar los registros duplicados) y generaría inconsistencias. Además, se puede perder mucho tiempo tratando de determinar por que algunos requerimientos son solicitados a la antigua dirección del servidor, debido a algún cambio mal realizado.

3.2.4. Registros CNAME

Los registros CNAME no deben coexistir con otros datos, por ejemplo si *jose.urzua.cl.* es un alias a *andres.urzua.cl.* no se deberían tener registros MX para *jose.urzua.cl.*, o registros de tipo A o TXT. Especialmente, no se deben combinar registros CNAME con registros NS:

```
jose.urzua.cl.    86401  IN  NS      ns.urzua.cl.
                  86401  IN  NS      ns2.urzua.cl.
                  86401  IN  CNAME   andres.urzua.cl.
andres.urzua.cl.  86401  IN  A       192.80.24.41
```

Generalmente estos errores se ven en administradores con poca experiencia, que desean que su nombre de dominio sea también un nombre de host. Sin embargo, los servidores DNS (como BIND) verán estos registros CNAME y no agregarán otros registros para ese nombre. Como no se permite la coexistencia de los CNAME con otros registros, los registros NS serán ignorados. Si se desea que el nombre de dominio sea también un nombre de host, se debe realizar de esta manera:

```
jose.urzua.cl.    86401  IN  NS      ns.urzua.cl.
                  86401  IN  NS      ns2.urzua.cl.
                  86401  IN  A       192.80.24.41
andres.urzua.cl.  86401  IN  A       192.80.24.41
```

Los registros CNAME son útiles y recomendados para los nombres generalizados de los servidores, tales como *ftp* para un servidor FTP, *www* para un servidor web, *news* para un servidor de usenet news. No se debe olvidar borrar los registros CNAME asociados a un host que ya fue borrado. Si se mantienen estos CNAMEs, serán sólo recursos perdidos.

No se deben usar los registros CNAME en combinaciones con RRs que apuntan a otros RRs como MX, CNAME, PTR y DNS. Por ejemplo:

```
jose.urzua.cl.    86401  IN  MX      mail.urzua.cl.
mail.urzua.cl.    86401  IN  CNAME   andres.urzua.cl.
andres.urzua.cl.  86401  IN  A       192.80.24.41
```

En el RFC 1034[3], sección 3.6.2 se dice que esto no se debe realizar, y en el RFC 974[7] se establece explícitamente que los registros MX no deberían apuntar a un alias definido por un CNAME. Esto resulta en direccionamientos innecesarios al acceder a estos datos, y los resolvers y servidores DNS necesitarán trabajar más para poder obtener una respuesta. Peor es el caso de los CNAME que apuntan a otro registro CNAME, en esos casos las resoluciones podrían fallar si algún resolver no es capaz de manejar de manera adecuada estas respuestas.

3.2.5. Registros MX

Es recomendable agregar a cualquier host un registro MX, aunque este apunte al mismo host. Algunos servidores de correo almacenarán en cache los registros MX, pero siempre necesitarán la presencia de un registro MX antes de enviar un email. Si para un sitio no existe un registro MX, entonces cualquier email resultará sólo en uno o más requerimientos de un resolver, ya que la respuesta para un requerimiento MX a menudo contiene la dirección IP del host MX.

Se deben agregar registros MX para los hosts aunque estos no estén destinados a enviar o recibir emails. Si existe algún problema con estos hosts, alguna persona podría tratar de enviar erróneamente un email a direcciones como *hostmaster*, *postmaster* o *root*, sin antes chequear que este host está o no habilitado para recibir emails. Si se agrega un registro de tipo MX, entonces el email será redireccionado a la persona o computador correcto. Si no se agrega, el email puede quedar encolado por horas o días hasta que el emisor sea notificado que debe reenviarlo.

3.2.6. Registros Wildcard

Un error común es pensar que un registro *wildcard* para un registro MX se aplicará para todos los host definidos en la zona. Un wildcard MX se aplicará solamente a los nombres que no están agregados en la zona, por ejemplo:

```
urzua.cl.          86401  IN  NS      ns.urzua.cl.
                   86401  IN  NS      ns2.urzua.cl.
                   86401  IN  A       192.80.24.41
andres.urzua.cl.   86401  IN  A       192.80.24.41
*.urzua.cl.        86401  IN  MX      5 mail.urzua.cl.
```

Un email para *andres.urzua.cl.* será enviado a él mismo para que sea despachado. Solamente emails para *paola.urzua.cl* o cualquier otro nombre de host que no esté en los nombrados arriba será enviado al registro MX.

Los registros wildcard pueden ser malos, ya que pueden generar operaciones exitosas cuando estas deberían fallar. Por otro lado, pueden ser buenos cuando se tiene un gran número de hosts los que no están directamente conectados a Internet (por ejemplo detrás de un firewall) y por razones políticas o administrativas es muy difícil tener registros MX para cada host, o en una situación en que se deban forzar que todas las direcciones de emails sean escondidas detrás de uno o más nombres de dominios. Para este caso, se debe dividir la información del DNS en dos partes, una interna y otra externa. La parte externa tendrá solamente unos pocos hosts y registros MX explícitos, y uno o más registros wildcard MX para cada dominio interno. Internamente la información del DNS debe ser completa, con todos los registros MX explícitos y no wildcards.

En la actualidad, la empresa Verisign Inc.[1] desarrolló un servicio llamado *Site Finder*, el cual utiliza la funcionalidad de los wildcard para dar resultados exitosos cuando un cliente solicita un nombre de dominio inexistente de segundo nivel para los TLDs .COM y .NET. Como resultado, el cliente recibía un mensaje de error en su navegador de Internet, con la implementación de Site Finder, el cliente recibe un sitio web que entrega ayuda y enlaces a los posibles destinos que solicita el cliente.

3.2.7. Errores de Delegación y Autoridad

Se recomienda tener al menos dos servidores de nombre para cada dominio, mientras más es mejor. También se recomienda tener servidores secundarios fuera de la red del primario, si los servidores secundarios no están bajo el control de los administradores del servidor primario, se recomienda revisar periódicamente que la información de la zona esté actualizada. Además, las respuestas de estos servidores siempre deben ser autoritativas.

Un error en la delegación típico se puede ver en este ejemplo:

```
urzua.cl.          86401  IN  NS      ns.urzua.cl.
                   86401  IN  NS      ns.paola.com.
```

En este caso, el nombre de dominio *urzua.cl.* es un nuevo dominio que está siendo creado, y *ns.urzua.cl.* será configurado con el servicio DNS para la zona de este dominio. Pero por otro lado, no existe la seguridad que el administrador del host *ns.paola.com.* configurará correctamente el servicio de servidor DNS secundario para el dominio, y *ns.paola.com.* tampoco tiene información sobre el dominio *urzua.cl.*, así todas las delegaciones o requerimientos que lleguen para el dominio al servidor secundario generarían un error conocido como *´lame delegation´*.

En algunos casos, los servidores de nombres son cambiados a otro host o borrados de la lista de los servidores secundarios. Desafortunadamente y debido a que los RRs NS son almacenados en cache, otros servidores seguirán pensando que un host es secundario para el dominio aún después de que el servidor a dejado de prestar esos servicios. Como recomendación para prevenir el error

[1] http://www.verisign.com

de *lame delegation*, se aconseja mantener el servicio de DNS en el servidor que cambiará, por lo menos por el tiempo en que la información volverá a ser refrescada.

Cuando un servidor primario o secundario es modificado o borrado, la coordinación entre los administradores involucrados toma un tiempo considerable. Cuando se modifica un servidor primario, se debe asegurar que todos los servidores secundarios sean actualizados y recarguen la información desde el nuevo servidor. Cuando se actualiza un servidor secundario, se debe asegurar que los registros en la zona padre sean modificados.

Los RRs de tipo NS nombrados en la zona padre, deben ser los mismos que se nombran en las zonas de los servidores secundarios. No se deberían tener demasiados, al menos deberían ser 2 y como máximo 7[2], ya que la administración será una tarea más compleja y sólo es necesario tener cerca de 7 para dominios muy populares y requeridos, como los dominios de primer nivel.

3.3. Problemas de Operación

Los problemas enunciados en esta sección tienen relación con la operación de un servidor de nombres con el software BIND. Para un correcto funcionamiento, no sólo los datos enunciados anteriormente deben estar sin errores, también se debe tener especial cuidado en como está operando el servidor.

3.3.1. Números Seriales

Cada zona tiene un número serial asociado a ella. Este número es usado para mantener un registro de quien tiene los datos de la zona más actualizados. Si sólo es servidor primario de la zona tiene el número serial más grande, los servidores secundarios solicitarán al primario una nueva copia de la zona.

No se debe olvidar cambiar el número serial de la zona cada vez que se actualicen los datos de ella. Si no se hace, los servidores secundarios no transferirán la nueva información de la zona, por

lo que un incremento automático del número serial por medio de un software es una buena idea (mientras se mantenga el formato anterior).

Si por algún error el número serial de la zona es incrementado en un número demasiado grande, y se desea reiniciar el valor a un valor más pequeño, se debe usar el siguiente procedimiento:

Tomar el número serial incorrecto y sumarle 2147483647 a él. Si el número es mayor que 4294967296, se debe restar 4294967296. Recargar la zona con este número resultante y esperar 2 tiempos de refresco para que la zona se propague a todos los servidores.

Se debe repetir lo anterior hasta que el número serial sea más pequeño que el nuevo número que se desea. Finalmente, se cambia el número serial de la zona al deseado.

Este procedimiento no funcionará correctamente en los servidores que tenga una versión antigua de BIND (4.8.3 o menor). En estos casos se debe contactar directamente a los administradores de estos servidores para que detengan el servicio de DNS, eliminen los archivos de zona almacenados y reinicien el servicio.

3.3.2. Archivo de Zona

Existen recomendaciones útiles para ayudar a estructurar los archivos de zonas de los dominios. Siguiendo estas indicaciones se podrá identificar los errores y así evitarlos a futuro.

Se debe ser consistente en los datos que se ingresan en los archivos de zona, si se está definiendo la zona del dominio *urzua.cl.*, no se deben usar las siguientes definiciones:

```
jose              86401  IN  A       192.80.24.30
andres.urzua.cl.  86401  IN  A       192.80.30.24
paola             86401  IN  A       192.80.33.24
                  86401  IN  MX      mail.urzua.cl.
```

Se recomienda usar todos los nombres de dominios como FQDNs (Fully Qualified Domain Names)[2] o usar nombres incompletos a los cuales se les anexará automáticamente el dominio que se está definiendo, pero no mezclar ambos. O usar los nombres completos al lado derecho y al lado izquierdo usar nombres relativos, pero sobre todo se debe ser consistente.

Para separar elementos dentro de la zona, se recomienda utilizar tabuladores y tratar de mantener columnas alineadas. Con esto, es más fácil la tarea de identificar campos que están faltando. También, se debe recordar que no es necesario repetir el nombre del host cuando se están definiendo múltiples RRs para el mismo host, idealmente se deben mantener todos los RRs de un mismo host juntos en el archivo de zona.

Siempre se debe tener presente el dominio que se está definiendo en la zona, si no se agrega el caracter punto al final del nombre que se está definiendo, este no será reconocido como FQDN y se agregará al final de este nombre el nombre del dominio origen de la zona. Se recomienda realizar cuidadosas revisiones de los nombres que se están definiendo, sobre todo en los archivos de zona del dominio *in-addr.arpa*.

Finalmente, se debe ser muy cuidadoso en los registros que utilizan paréntesis, como los RRs de tipo SOA y WKS. El software BIND no es demasiado flexible en la interpretación de estos RRs.

3.3.3. Verificación de Datos

Después de alguna actualización en los datos, se recomienda verificar que estos datos sean los correctos mediantes consultas a un resolver por medio de alguna herramienta como *dig*, después de realizado el cambio. Unos pocos segundos que se utilicen en esta revisión pueden salvar horas de dificultades, pérdidas de emails o errores generales. También se debe estar pendiente del registro que va quedando en el sistema cada vez que se reinicia el servicio de DNS (revisión de logs), generalmente quedan registrados cualquier error encontrado en algún archivo de zona u otro.

Capítulo 4

Diseño de la Herramienta

4.1. Introducción

En este capítulo, se presenta todo el proceso de diseño que se realizó para construir la herramienta que permite realizar el estudio para los nombres de dominio. En primer lugar, se muestra la lista de características que se deben revisar tanto en la configuración de los servidores como en las respuestas que ellos entregan. Luego, se explican las decisiones previas que el autor debió tomar para concretar el diseño, para luego pasar a describir las plataformas de hardware y software que se utilizan y finalizar con una completa descripción del diseño de la herramienta, con su correspondiente diagramas de flujo y funcional.

4.2. Lista de Características a Revisar

Luego de revisiones detalladas de los RFCs relacionados al funcionamiento del DNS, del análisis de los estudios y herramientas que actualmente se utilizan para medir el estado del DNS, se determinaron ciertas características que son de mucho mayor importancia que otras, que son recomendaciones. Para diferenciar las características importantes de las recomendaciones, se agrega

la letra N (necesarias) o R (recomendaciones), según corresponda. El listado de estas características y recomendaciones son:

1. **Todos los servidores de nombres del dominio deben responder. (N)** Si un servidor de nombres del dominio no responde, podría generar que el dominio sea inalcanzable para algunos requerimientos, sobre todo para el caso en que existe sólo 1 servidor de nombres.

2. **Todos los servidores deben tener un reverso configurado correctamente. (N)** Los servidores de nombres del dominio son manejados como nombres de *host* dentro de las respuestas de los demás servidores de nombres, de no existir un reverso para el nombre del servidor, este será inalcanzable.

3. **Todos los servidores de nombres del dominio deben responder con autoridad. (N)** Cuando un servidor DNS de un nombre de dominio no responde de manera autoritativa existirán requerimientos para el dominio que fallarán, además ese servidor no cumplirá con el objetivo de que sea una fuente de redundancia de información para el nombre de dominio.

4. **Los registros NS definidos no deben ser del tipo CNAME. (N)** El usar CNAMES obliga a realizar una nueva resolución al nombre del servidor NS, que en el caso de ser un *glue record* podría generar errores en la resolución.

5. **Registro MX debe ser un nombre de *host* y no una dirección IP. (N)** Está establecido que el registro MX debe ser un nombre de *host* y no una dirección IP.

6. **Registros MX deben tener un reverso bien configurado. (N)** Como el registro MX es un nombre de host, para lograr una correcta resolución de la máquina que debe recibir el email, debe existir el reverso bien configurado para el registro MX.

7. **Todos los servidores DNS del nombre de dominio deben responder con los mismos servidores de nombres que responde el servidor DNS de primer nivel. (R)** De cumplirse, indicaría que existe uniformidad entre la información que se mantiene entre todos los servidores de nombres del dominio y el servidor de nombres de más alto nivel. Para el caso de los dominios .CL, por ejemplo los servidores de nombres del dominio *gobierno.cl* deben responder con los mismos servidores de nombres que responde el servidor encargado del nombre de dominio .CL. De no cumplirse, se corre el riesgo de que el servidor de nombres no mencionado en alguno de los demás pueda ser inalcanzable para los requerimientos que se generen para el nombre de dominio.

8. **El número de servidores DNS para el nombre de dominio debería estar entre 2 y 7. (R)** Por motivos de redundancia, es muy importante mantener más de un servidor de nombres para el dominio, pero por velocidad de actualización y disminución del riesgo de información no uniforme, se recomienda mantener entre 2 y 7 servidores de nombres.

9. **Las respuestas de los servidores de nombres no deben ser recursivas. (R)** Un servidor que acepte consultas recursivas además de realizar mayor trabajo para responder un requerimiento, esta bajo el riesgo de ser atacado por *cache poisoning*, con lo cual podría entregar respuestas erróneas para algunos requerimientos.

10. **Las direcciones IP de los servidores DNS de un dominio deben estar en clases separadas. (R)** Por motivos de robustez ante caídas de enlace para un rango de direcciones IP, es recomendable que los servidores de nombres de un nombre de dominio estén en clases IP distintas.

11. **Versiones del software de DNS en los servidores de nombres deben ser distintas. (R)** De esta manera, si se descubre alguna vulnerabilidad en el software que se está usando no todos los servidores se verán afectados.

12. **Todos los servidores de nombres deben responder con el mismo número serial para la información del dominio. (R)** El número serial del registro SOA en las zona del dominio, identifica la versión de esa información, para que los servidores respondan con la misma información, todos deben tener el mismo número serial para el registro SOA.

13. **En el registro SOA debe estar el nombre del servidor primario del dominio. (R)** El primer parámetro en el registro SOA debe ser el nombre del servidor DNS primario del dominio, ya que este registro es la principal fuente de información del nombre de dominio.

14. **El formato del número serial del SOA debería tener el formato AAAAMMDDnn, con nn el número de versión de la zona de ese día. (R)** Este formato, es recomendable debido a que todo cambio que se realice en la zona generará un nuevo número serial que será siempre mayor al que existía.

15. **Los tiempos del registro SOA deberían cumplir con: reintento menor que refresco, refresco menor que ttl, ttl menor que expiración. (R)** De esa manera, todas las actualizaciones de la información se realizarán de la manera recomendada, y los nombres estarán acorde a lo que ellos representan.

16. **El tiempo de refresco en el registro SOA debe estar entre 3600 a 7200 (el RFC 1912 dice que debe estar en 1200 y 43200). (R)** Es importante que el tiempo de refresco no sea demasiado pequeño, pues generaría demasiado tráfico innecesario en el caso de que la información no cambie muy frecuentemente, en el caso de tener un tiempo demasiado alto, los cambios en la información de dominio se propagarán muy lento.

17. **El tiempo de reintento en el registro SOA debería estar entre 120 a 7200. (R)** Además, debería ser menor que el tiempo de refresco para que los reintentos para obtener el número serial de la información, ocurran antes que se cumpla el tiempo de refresco.

18. **El tiempo de expiración en el registro SOA debería estar entre 1209600 a 2419200 (2 a 4 semanas). (R)** Por las mismas razones explicadas anteriormente, este tiempo debería estar entre los recomendados. Además, debería ser mayor al TTL.

19. **El tiempo de vida de la información (TTL) debería estar entre 86400 a 432000 (1 a 5 días, RFC 1912). (R)** Al igual que las razones para los demás tiempos del registro SOA, este tiempo debería estar entre los recomendados y ser menor que el tiempo de expiración de la información.

20. **No debería ser CNAME el registro WWW. (R)** Uno de los registros más consultados es el WWW, si es un CNAME se estaría realizando resoluciones redundantes, lo aconsejable es que sea un registro A directamente.

21. **Deberia existir un registro MX para el nombre de dominio. (R)** De no existir, todos los emails que se generen para el dominio los recibirá el registro *ADDRESS*.

22. **Registros MX no deben ser CNAMES. (R)** Si están definidos como CNAME, se estaría obligando a realizar una nueva resolución del nombre del servidor de email, el cual si es un *glue record* podría generar errores en la resolución.

23. **Debería existir más de un registro MX. (R)** Por motivos de redundancia debería existir más de un registro MX.

Este listado de características por revisar para los servidores DNS de un nombre de dominio, se consideran suficientes y adecuadas para realizar un primer estudio del estado del DNS. Cabe mencionar que estas características se han recopilado de los variados RFCs que describen el correcto funcionamiento del DNS, de los estudios ya realizados por otras instituciones para otros dominios y de herramientas existentes para el diagnóstico de DNS. Además, está la experiencia del personal de soporte técnico de NIC Chile en donde se reportan y solucionan las dudas de los administradores de servidores DNS de los nombres de dominios registrados bajo .CL.

Probablemente para futuros estudios se agreguen o saquen características, dependiendo de los resultados y experiencia de haber realizado esta investigación por primera vez.

4.3. Decisiones Previas

Uno de los objetivos de la herramienta a diseñar, es que se puedan realizar estudios periódicos comparables en el tiempo. De esta manera, lo más adecuado es mantener los reportes generados en una base de datos con una estructura que permita registrar la completa evaluación de los servidores DNS de los nombres de dominio.

Para identificar a qué estudio pertenecen los datos recopilados, el nombre de la base de datos tendrá un sufijo que indicará la fecha en la que se realizó el estudio. Por ejemplo, para un estudio realizado el 23 de Diciembre del 2003, el nombre de la base de datos será: INSPECTORDNS20031223. De esta manera, se mantendrán nombres diferentes para la base de datos de cada estudio a realizar, teniendo la posibilidad de realizar las mismas consultas para distintos estudios con sólo cambiar el nombre de la base de datos. En la base de datos, se almacenará la información completa de los nombres de dominios que se están analizando junto con los servidores de nombres registrados en NIC Chile para el servicio de DNS del nombre de dominio. También se almacenarán las características a evaluar en el estudio, junto con los resultados obtenidos al evaluar cada dominio.

El desarrollo de la herramienta será con el lenguaje de programación Perl y MySQL como motor de base de datos. También se utilizará principalmente el módulo NET::DNS de Perl, el cual permitirá realizar consultas a servidores que proveen el servicio de DNS. Dentro de la ejecución de la herramienta, se poblará la base de datos con las respuestas que se obtengan de los servidores y una vez que se termine de analizar para todos los servidores, se generarán archivos con formato HTML que mostrarán los resultados del estudio.

Comúnmente, en una zona grande como la de un TLD, se tendrá que un servidor de nombres atenderá los requerimientos de varios nombres de dominios. Con el objetivo de ahorrar tiempo de ejecución de la herramienta, y debido a que se aplicarán las mismas consultas a los servidores de nombres de todos los nombres de dominio, se implementó una estructura que mantiene los nombres de los servidores que no están respondiendo y de esa manera si es necesario consultarlo nuevamente se lee la respuesta desde esta estructura y no se intenta realizar la consulta nuevamente, la que puede costar entre 10 y 15 segundos. Esta misma estructura se utiliza para cuando se quiere obtener los reversos de los servidores, la cual se ejecutará solo una vez dentro de toda la ejecución.

A medida que pase el tiempo y el software que provee los servicios de DNS se adecúe a las nuevas necesidades, será necesario ir agregando o sacando características a revisar, por lo que es

necesario que la herramienta esté construida de tal manera que agregar una nueva evaluación no sea una tarea muy compleja. Con esto, también se permite una colaboración externa de cualquier persona que desee contruibir en mejorar la herramienta.

La herramienta quedará a libre disposición en Internet, con su documentación correspondiente para que cualquier persona que desee usarla lo pueda realizar de manera simple. Todo lo necesario para utilizar la herramienta es provisto por software con licencia GNU General Public License (GPL) en el caso de MySQL y Artistic License[1] en el caso de Perl. Ambas licencias son consideradas de libre distribución, por lo que se pueden hacer implementaciones para diversas plataformas fomentando la distribución de esta herramienta.

El desarrollo y ejecución de la herramienta será en un computador con las siguientes características de hardware y software:

- AMD Athlon(TM) XP 2200+, de 1800 MHz
- 1 Disco duro de 60 GB, marca Maxtor modelo 6Y060L0
- 1 GB de memoria RAM
- Sistema Operativo Linux Red Hat 9.0
- Motor de Base de Datos MySql 4.0.16
- Lenguaje Perl, versión 5.8.0

El lenguaje Perl, esta complementado con módulos específicos para realizar consultas a servidores DNS (Net::DNS), análisis de zonas (DNS::ZoneParse), conexión a bases de datos (DBI) y generación de gráficos (GD::Graph).

[1] Licencia compatible con GNU GPL, la versión original de esta licencia es muy vaga respecto a si es una licencia de software libre.

4.4. Diseño de la Herramienta

Para que la herramienta sea perdurable en el tiempo, es necesario que su diseño sea bastante modular y documentado. En particular, se tendrán bibliotecas encargadas de cada proceso dentro de la ejecución, como es la creación de la base de datos, consultas a servidores externos y generación de reportes.

Para explicar más a fondo el diseño, a continuación se muestrán los diagramas de arquitectura, de flujo de datos y modelo de datos de la herramienta, con su correspondiente descripción.

4.4.1. Diagrama de Arquitectura

En la figura 4.1 se puede apreciar la arquitectura de la herramienta. En esta figura se puede distinguir un funcionamiento local, con el computador en donde se está ejecutando la herramienta, el cual interactua con una base de datos local en donde crea y actualiza los registros de acuerdo al resultado de la ejecución. Tambien se aprecia un archivo de zona, el cual es utilizado para obtener los nombres de dominios y los servidores que prestan el servicio de DNS para él.

Por otro lado, se puede apreciar una interacción con servidores de nombres externos. Estos servidores recibirán consultas DNS enviadas por Internet para el dominio correspondiente, y de paso entregarán las correspondientes respuestas, las que generarán actualizaciones en la base de datos de la herramienta.

4.4.2. Diagrama de Flujo

En la figura 4.2 se muestra el flujo de los datos al ejecutar la herramienta.

La ejecución comienza creando la base de datos según el modelo de datos de la figura 4.3. Luego, se pobla la base de datos con las características a analizar y la información de los dominios,

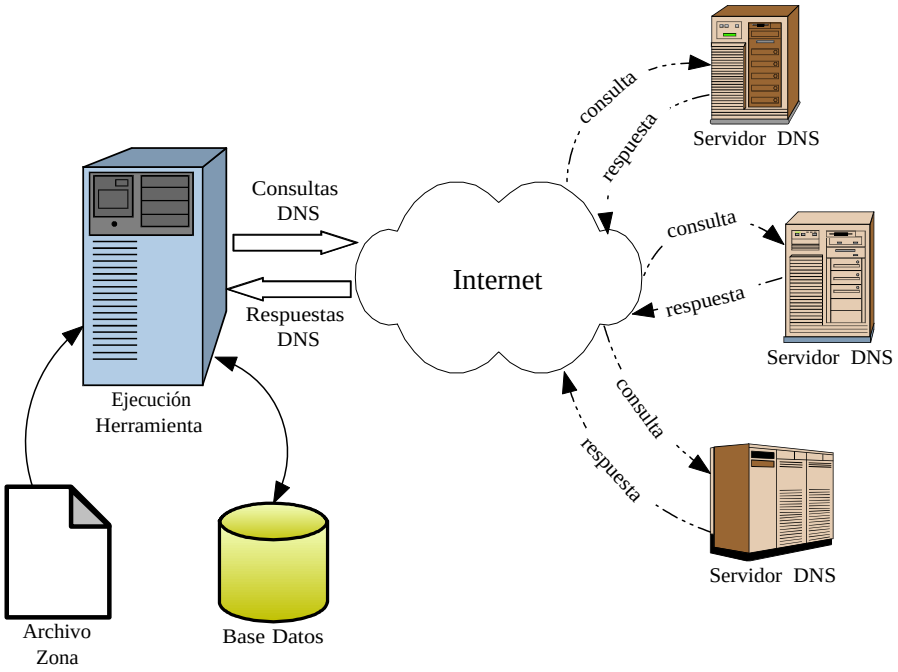

Figura 4.1: Diagrama de Arquitectura.

que se obtiene desde la zona correspondiente. Esta información involucra el nombre del dominio y los servidores de nombres registrados en NIC Chile, en caso de estar presentes las direcciones IP de los servidores (para el caso de los glue records), estas también se almacenarán en la base de datos.

Una vez que se tiene poblada la base de datos con la información a analizar, se comienza con la evaluación de los dominios, aplicando cada función con su correspondiente característica. Cuando se tenga el resultado de la evaluación, se procede a insertar en la base de datos el resultado.

Finalmente, cuando se evaluaron todas las características para todos los dominios, se procede a la generación de reportes en formato HTML con sus correspondientes gráficos. Estos reportes se dejarán disponibles en Internet.

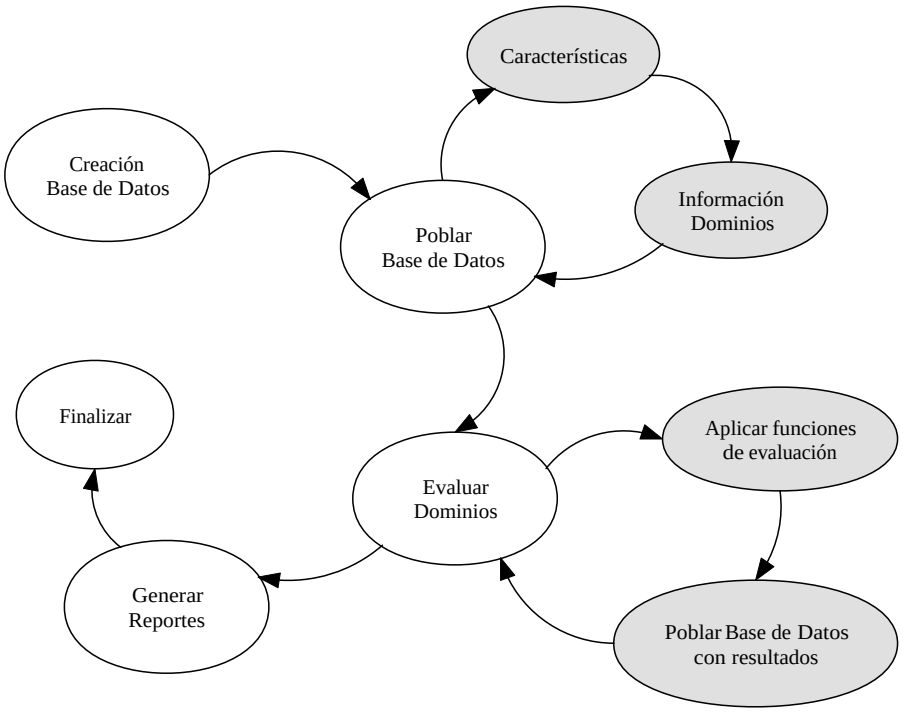

Figura 4.2: Flujo de Datos en la Herramienta.

4.4.3. Modelo de Datos

El modelo de datos, que se utilizará para mantener los resultados de la ejecución de la herramienta se puede apreciar en la figura 4.3. En este modelo, se reflejan los datos correspondientes a un dominio y sus servidores de nombres, para los cuales se almacenan su(s) direción(es) IP. También se puede apreciar que se almacenarán las características que se están evaluando y su correspondiente evaluación.

En la entidad DOMINIO, se almacenarán todos los nombres de dominios presentes en la zona que se analizará. Para cada dominio se almacenará su nombre y contacto que aparece en la zona, además se asignará un identificador único, al momento de insertarlo en la base de datos.

Los datos de los servidores de nombres se almacenarán en la entidad SERVIDOR, la cual mantendrá el nombre del servidor y un identificador. La asociación entre un dominio y sus servidores de nombres, se mantiene en la entidad DOMINIO_SERVIDOR, que mantendrá en sus registros los

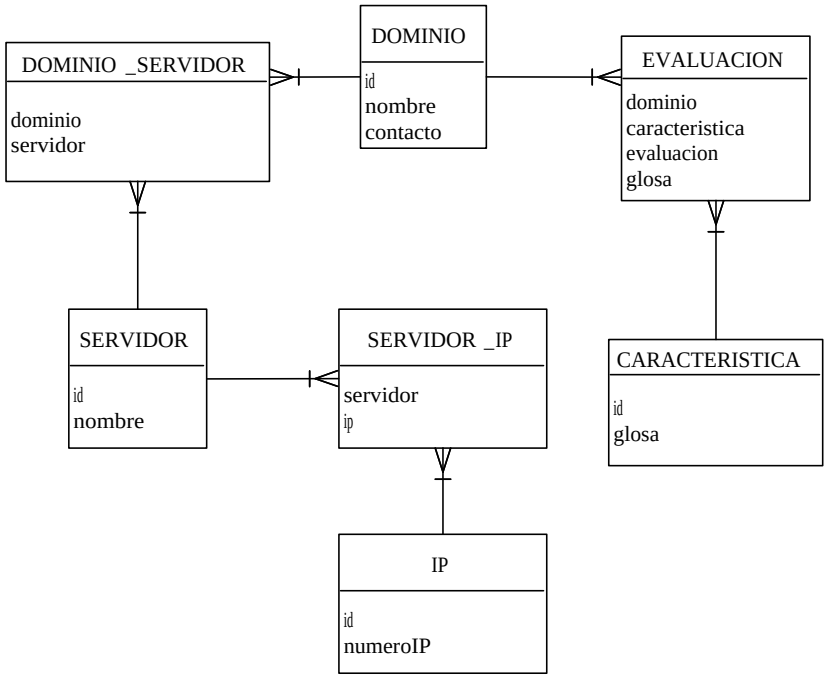

Figura 4.3: Modelo de Datos.

identificadores de dominio con sus servidores asociados. Como un servidor de nombres puede tener más de una dirección IP, estas se mantendrán en la entidad IP, en donde se almacenará la dirección IP junto con un identificador de ella, y la asociación con el servidor de nombres se mantendrá en la entidad SERVIDOR_IP.

Las características que se evaluarán en esta versión de la herramienta se almacenarán en la tabla CARACTERISTICA, en donde se tendrá un identificador de la característica junto con una glosa descriptiva. Además, se tendrán dos glosas, una en caso de que falle la evaluación y en caso que resulte exitoso.

Del resultado de evaluar una característica para un dominio y sus servidores de nombres, se poblará la tabla EVALUACION, en donde se tendrá registrado, el dominio que se evaluó, la característica que se aplicó, el resultado de la evaluación y una glosa del resultado.

Capítulo 5

Implementación

5.1. Introducción

Luego de estudiar el DNS y los problemas que lo pueden afectar y especificado el diseño de la herramienta, se da paso a la implementación, tema que se expone en este capítulo. Acá se muestran las descripciones de los módulos y programas que componen la herramienta, el como se abordó la implementación de las evaluaciones para cada una de las características y extractos del código fuente de algunas partes esenciales de la herramienta.

5.2. Módulos y Programas Útiles

Para la implementación de la herramienta, se desarrollaron 2 módulos:

- **LibInspector.pm**: Módulo que posee todas las funciones de conexión a la base de datos, ya sea para insertar, actualizar y seleccionar registros. También contiene funciones útiles que

son requeridas en el programa principal.

- **ConfigInspector.pm**: Módulo que mantiene las variables de configuración de la herramienta. En este módulo se definen las sentencias SQL necesarias para interactuar con la base de datos, las variables de conexión a la base de datos, de configuración de la evaluación y las variables necesarias para generar los reportes de la evaluación, en formato HTML.

Además de los módulos descritos, se tienen 3 programas principales que ejecutan todas las acciones de la evaluación. El más importante es el llamado *inspector.pl*, el cual se encarga de todas las tareas de la evaluación, como es la creación de la base de datos, leer e insertar la información de los nombres de dominios a la base de datos y aplicar las evaluaciones definidas para los nombres de dominio.

Luego de realizada la evaluación, se deben obtener los reportes necesarios en formato HTML, para ello, se implementó el programa *reporteInspector.pl*, el cual recorre la tabla de evaluaciones de la base de datos, obteniendo los resultados de la evaluación de cada característica y además genera los gráficos y archivos en formato HTML correspondientes. Para la obtención de reportes detallados por cada nombre de dominio, se implementó el programa *reporteDetalle.pl*, el cual actúa como un *CGI*, que solicita un nombre de dominio y luego genera una página con un completo detalle de la evaluación realizada para el nombre de dominio.

5.3. Implementación de Evaluación de Características

De acuerdo al diagrama de flujo presentado en el subcapítulo 4.4.2, luego de insertar en la base de datos la información de los nombres de dominios y sus servidores de nombres en la zona analizada, se procede a evaluar cada una de las características para cada nombre de dominio y sus servidores.

Se debe considerar que la evaluación de cada característica se implementó de distinta manera, de acuerdo a lo que se desea evaluar. A continuación, se presenta el listado de las características y como se implementó su evaluación:

1. Completitud de Servidores: Para esta evaluación, es necesario conocer las respuestas de todos

los servidores de nombres del dominio. Se realiza dentro de un ciclo, en el cual se realiza una consulta de tipo DNS, preguntando por todos los registros de tipo **NS** que conocen los servidores de nombres para el dominio en evaluación. Una vez que finaliza el ciclo, se comparan las respuestas de cada servidor, de ser idénticas entre ellas y también con la información registrada en la zona padre (para esta tesis corresponde a la zona de .CL) la evaluación de la característica estaría correcta, de lo contrario se marcaría como error.

2. Servidores Activos: Si bien es una consecuencia de la evaluación anterior, es necesario identificar que servidores están activos, alcanzables y que respondan consultas DNS. Cada servidor se va identificando como activo de acuerdo si respondió para la evaluación anterior. Si todos los servidores responden se evalúa positivamente esta característica, de lo contrario se marca como error.

3. Reversos de Servidores bien configurados: Esta evaluación, se realizó por medio de un ciclo que itera sobre cada servidor de nombres del dominio, obteniendo las direcciones IP del servidor y aplicando una consulta de 'resolución' para cada dirección. Si el nombre de servidor obtenido con esta última consulta es igual al servidor de nombres que se está analizando, se marca como positiva la evaluación para este servidor de nombres. Si las evaluaciones para todos los servidores es positiva se marca la característica como positiva, de lo contrario será un error.

4. Número de servidores: Para esta evaluación, se obtiene el número de servidores de nombres registrado en la zona padre, si este número está dentro de los recomendados se marca como positiva la evaluación de lo contrario será error.

5. Respuestas con Autoridad: Para ésta evaluación se utiliza la consulta realizada en la primera característica, obteniendo los *headers* de la respuesta y analizando si vienen con 'autoridad', si todos los servidores responden con autoridad se marca como correcta, de lo contrario será error.

6. Respuestas no deben ser recursivas: Utilizando la respuesta a la primera característica, se obtienen los *headers* de la respuesta y se analiza si vienen con disponibilidad de recursividad. Si todos los servidores responden sin recursividad la evaluación será positiva, de lo contrario se marcará como error.

7. Registros NS no deben ser de tipo CNAME: Para esta evaluación, se consulta por los registros de tipo NS en cada servidor, y se analiza si estan definidos como CNAME. Si todas las respuestas indican que no son de tipo CNAME, la evaluación será correcta, de lo contrario será error.

8. Direcciones IP en subredes separadas: Cuando se tiene la base de datos poblada con la información de las direcciones IP de los servidores de nombres del dominio, se obtienen dichas direcciones y se analiza si sus 3 primeros octetos son iguales. De ser así, tendriamos direcciones IP que están en la misma subred, lo que marcaría como negativa la evaluación de

esta característica, basta con que una dirección IP esté en una subred distinta para que la evaluación sea positiva.

9. Distintas versiones de Software DNS: Esta evaluación se realizó enviando consultas de tipo DNS, con algunas clases y definiciones especiales, para tratar de obtener que versión de software DNS está funcionando en el servidor. De ser una versión diferente a las demás, se marcará la característica como positiva, de lo contrario será error. Cabe señalar, que el tipo de consulta utilizado no está especificado en ningún RFC, y las respuestas que se pueden obtener son 'configurables' por los administradores de dichos servidores.

10. Uniformidad de número serial: Para esta evaluación, se realiza una consulta DNS solicitando el registro SOA para el nombre de dominio en cada servidor. De la respuesta de cada servidor, se obtiene el número serial y finalmente se comparan los números seriales obtenidos. De ser iguales, la evaluación será positiva, de lo contrario será un error.

11. Nombre de servidor primario en el registro SOA: Utilizando la consulta realizada para la evaluación anterior, se obtiene el parámetro denominado MNAME, el cual debe coincidir con uno de los servidores de nombre declarado en la zona padre del nombre de dominio. De coincidir, la evaluación será positiva, de lo contrario se marcará como error.

12. Formato del número serial: Utilizando la consulta por el registro SOA, se obtiene el número serial y se analiza su formato. Se trata de traducir en una fecha, tomando los 4 primeros caracteres como un año, los 2 siguientes como un mes y los dos siguientes como un número de día. Así, se aplican las reglas concernientes a los meses en donde el valor debe estar entre 1 y 12, a los dias en donde el valor debe estar entre 1 y 31. Para el año, se obtiene el valor del año actual en el sistema y se analiza si el año obtenido del número serial está dentro de 5 años más o menos que el obtenido del sistema. Si las fechas cumplen con las características mencionadas, la evaluación se marca como positiva, de lo contrario será error.

13. Relación de los tiempos del registro SOA: Utilizando la consulta realizada por el registro SOA, se obtienen los tiempos de refresco, reintento, expiración y tiempo de vida. Luego, se analiza que cumplan la relación reintento menor que refresco, refresco menor que tiempo de vida, tiempo de vida menor que expiración. De cumplirse, la característica será marcada como positiva, de lo contrario será error.

14. Tiempo de refresco del registro SOA debe estar dentro de los recomendados: De la evaluación anterior, se obtiene el tiempo de refresco y se analiza si está dentro de los recomendados. De cumplirse, la característica será marcada como correcta, de lo contrario será error.

15. Tiempo de reintento del registro SOA debe estar dentro de los recomendados: Se evalúa de la misma manera que el anterior.

16. Tiempo de expiración del registro SOA debe estar dentro de los recomendados: Se evalúa de la misma manera que el anterior.

17. Tiempo de vida del registro SOA debe estar dentro de los recomendados: Se evalúa de la misma manera que el anterior.

18. Registro WWW no debería ser de tipo CNAME: Para esta evaluación, se realiza una consulta de tipo DNS, preguntando a los servidores de nombres del dominio por el valor del nombre 'www.dominio.cl.', en donde dominio es el nombre de dominio que está en evaluación (.cl para el caso de la evaluación realizada en esta tesis). De encontrar respuesta a la consulta realizada, se analiza si el registro obtenido es de tipo CNAME, de serlo, la evaluación será negativa para esta característica, de lo contrario, será positiva.

19. Existencia de registro MX: Para esta evaluación, se realiza un consulta de tipo DNS, preguntando a los servidores de nombres de dominio por el valor del registro MX. De existir una respuesta completa, indicaría que existe por lo menos un registro MX para el nombre de dominio. De cumplirse, la evaluación será positiva, de lo contrario será negativa.

20. Registro MX no debe ser de tipo CNAME: Utilizando la consulta de la característica anterior, se analiza la respuesta para determinar si el registro obtenido es de tipo CNAME, de serlo la evaluación será negativa, de lo contrario se marcará como positiva.

21. Registro MX debe ser un nombre de Host: Para esta evaluación se utiliza la consulta realizada para obtener los registros MX, de la respuesta a esta consulta, se obtiene el nombre de servidor definido para el registro MX, y se analiza si cumple con las reglas de un nombre de host.

22. Más de un registro MX: Para esta evaluación, se utiliza la consulta realizada para obtener los registros MX, de la respuesta a esta consulta, se cuentan los servidores MX definidos, de ser más de uno, la evaluación será positiva, de lo contrario será un error.

23. Reversos de servidores MX bien definidos: De los nombres de servidores MX definidos para el dominio, se obtienen las direcciones IP asociadas, y para cada dirección se obtiene el nombre de servidor asociado. De coincidir con el nombre de servidor MX, la evaluación será positiva, de lo contrario será negativa.

5.4. Código de Funciones Principales

Dentro del código fuente que compone la herramienta, existen ciertas partes que son claves en el funcionamiento de la herramienta, como las consultas DNS realizadas. Por ejemplo, para obtener todos los registros NS asociados a un nombre de dominio, se utiliza el siguiente código:

```
my $res = Net::DNS::Resolver->new;
$res->nameservers($ser);
res->udp_timeout(3);
my $answer = $res->query($dom, 'NS');

if ($answer){
  foreach my $rr ($answer->answer) {
    push (@servidoresNS, lc($rr->nsdname).".") if ($rr->type eq "NS");
  }
}
else {
  $glosa[1] .= "<br>Error al consultar $ser: ". $res->errorstring ."<br
     >";
}
```

El código fuente anterior corresponde a parte de la evaluación de la primera característica (completitud de servidores de nombres), para lo cual, en primer lugar se declara un objeto 'resolver', identificado por la variable $res. A este resolver, se indica a que servidor de nombres se desea consultar, mediante el método 'nameservers', en donde la variable $ser tiene el nombre de servidor de nombres correspondiente. Luego, se asigna el tiempo de espera para esta consulta, mediante el método 'udp_timeout', para este caso se utiliza 3 segundos. Teniendo asignado el tiempo de espera y el servidor que se desea consultar, se realiza la consulta por los registros NS para el dominio identificado por la variable $dom, mediante la llamada del método 'query' del objeto resolver. La respuesta que retorna este método puede ser vacia en caso de algún error, en caso de no ser vacia se analiza para obtener los registros de tipo NS, los cuales se almacenan en un arreglo para su posterior análisis. En caso de que la respuesta sea vacía, se obtiene el mensaje de error desde el objeto 'resolver', mensaje que se almacena en la variable correspondiente (el elemento 1 del arreglo @glosa).

A continuación, se tiene un extracto del código fuente necesario para extraer los encabezados de las respuestas DNS, para poder determinar la autoridad y recursividad de las respuestas:

```
my $header = $answer->header;
if ($header->aa){
  $evaluacion[5] = 1 if ($evaluacion[5] != 0);
  $glosa[5] .= "<p>El servidor $ser responde con autoridad</p>";
} else {
  $evaluacion[5] = 0;
  $glosa[5] .= "<p>El servidor $ser responde SIN autoridad</p>";
}
```

```
if (!$header->ra){
  $evaluacion[6] = 1 if ($evaluacion[6] != 0);
  $glosa[6] .= "<p>El servidor $ser responde sin recursividad</p>";
} else {
  $evaluacion[6] = 0;
  $glosa[6] .= "<p>El servidor $ser responde con recursividad</p>";
}
```

Al llamar el método 'header' se obtienen los encabezados de la respuesta, los cuales son almacenados en la variable $header. Luego, se utilizan los métodos 'aa' y 'ra' para obtener la autoridad y recursividad de la respuesta, para finalmente asignar los mensajes y valores correspondientes de la evaluación de estas características.

Para saber si un registro está definido como CNAME, se realiza una consulta DNS para obtener el registro 'A' asociado al registro. Por ejemplo, para saber si el 'www' está definido como CNAME, se utiliza el siguiente código:

```
my $answerwww = $res->query("www.".$dom, "A");
if ($answerwww){
  foreach my $rr ($answerwww->answer) {
    if ($rr->type eq "CNAME"){
      $evaluacion[18] = 0;
      $glosa[18] .= "<p>Error, el registro WWW para $dom en $ser, ".
                    "áest definido como CNAME:<br><pre>".
                    $rr->string ."</pre></p>";
    }
  }
} else {
  $evaluacion[18] = 0;
  $glosa[18] .= "<p>Error, no pude determinar si el registro WWW áest ".
            "definido en $ser como CNAME: ". $res->errorstring. "</p>"
               ;
}
```

En este código, primero se realiza la consulta DNS por el registro de tipo 'A' para el host 'www.dominio.cl.", si se obtiene respuesta, se pasa a revisar el tipo del registro que viene en la respuesta, en caso de ser de tipo 'CNAME' se marca la evaluación y el mensaje correspondiente. En caso de no obtener respuesta se asigna el mensaje de error correspondiente.

Capítulo 6

Pruebas Realizadas

6.1. Introducción

Luego de realizada la implementación de la herramienta, se realizaron variadas pruebas para comprobar que cumple con las metas propuestas, estas pruebas y sus resultados son expuestos en este capítulo.

Todas las pruebas realizadas fueron sobre la zona del TLD .CL, ejecutándose sobre la plataforma descrita en la sección 4.3 del capítulo diseño.

6.2. Ambiente de Pruebas

Antes de ejecutar la herramienta, es necesario preparar un ambiente con todos los elementos necesarios para su correcta ejecución. Estos elementos son:

- Instalación Base de Datos: De acuerdo a lo especificado en el manual de usuario del sistema y en la sección 4.3, se debe crear una base de datos dentro del Motor MySQL. Para las pruebas realizadas se crearon utilizando las siguientes sentencias SQL dentro del *shell* MySQL:

```
mysql> create database INSPECTORDNS2004_01;
Query OK, 1 row affected (0.05 sec)

mysql> grant all privileges on INSPECTORDNS2004_01.* to inspector iden
Query OK, 0 rows affected (0.00 sec)

mysql>
```

  Con la ejecución de estas sentencias, se crea la base de datos y se entregan los permisos necesarios para el usuario *inspector* con clave de acceso *inspectorpass*.

- Configuración del Sistema: Luego de creada la base de datos es necesario asignar los valores correspondientes a las variables de conexión a la base de datos y la ruta de los archivos de zona, de creación de las tablas de la base de datos y las características que se evaluarán en el sistema.

- Archivo de Zona: El archivo de zona debe cumplir con el formato de zona del software Bind, además por lo menos debe tener permisos de lectura. Lo ideal es que el archivo de zona tenga la información mas reciente del dominio a evaluar.

Teniendo listo el ambiente de pruebas, sólo queda ejecutar la herramienta.

6.3. Ejecución

Durante la ejecución, la herramienta comenzará leyendo el archivo de creación de las tablas de la base de datos y ejecutará cada sentencia de creación. Luego, leerá el archivo de características a evaluar, para poblar la tabla correspondiente de la base de datos. A continuación, leerá el archivo de la zona, del cual obtendrá todos los nombres de dominios con los servidores de nombres asociados y los insertará en la base de datos. Finalmente comenzará la evaluación, recorriendo cada nombre de dominio y sus servidores DNS para los cuales se aplicarán las 23 evaluaciones especificadas en el archivo de características.

Mientras la herramienta aplica las evaluaciones, escribirá a la salida estándar mensajes sobre lo que está ejecutando, en particular se mostrará el nombre de dominio que se está evaluando e información relacionada con las consultas DNS.

6.3.1. Primera Evaluación

Esta primera evaluación se realizó el 22 de marzo del 2004, y la herramienta demoró 42 horas con 31 minutos y 24 segundos en evaluar 77293 nombres de dominio definidos en la zona de .CL, los cuales reciben el servicio de DNS desde un total de 12218 servidores de nombres.

El archivo que describe la zona de .CL utilizada, se obtuvo desde un servidor de NIC Chile el cual presta el servicio de DNS primario para .CL. Esta zona se genera 2 veces al día, y en ella se definen claramente y con el formato de Bind todos los nombres de dominios que tienen servidores de nombre bajo .CL.

Los resultados son los siguientes:

Nombre	Tipo	Evaluación OK	Evaluación Error	No Determinada
Servidores Activos	N	50589 (65.45 %)	0 (0.00 %)	26704 (34.55 %)
Reversos de Servidores	N	23004 (29.76 %)	42755 (55.32 %)	11534 (14.92 %)
Respuesta con Autoridad	N	43608 (56.42 %)	6981 (9.03 %)	26704 (34.55 %)
Análisis de CNAME para NS	N	50145 (64.88 %)	444 (0.57 %)	26704 (34.55 %)
MX es nombre de HOST	N	33362 (43.16 %)	5321 (6.88 %)	38610 (49.95 %)
Reverso de MX	N	6205 (8.03 %)	27480 (35.55 %)	43608 (56.42 %)
Completitud de Servidores	R	18646 (24.12 %)	58647 (75.88 %)	0 (0.00 %)
Número de servidores	R	70472 (91.18 %)	6821 (8.82 %)	(0.00 %)
No Recursividad	R	12072 (15.62 %)	38517 (49.83 %)	26704 (34.55 %)
Clases de direcciones IP	R	42358 (54.80 %)	34935 (45.20 %)	(0.00 %)
Versiones de Software DNS	R	28897 (37.39 %)	48396 (62.61 %)	(0.00 %)
Número Serial	R	51799 (67.02 %)	3506 (4.54 %)	21988 (28.45 %)
Nombre Servidor Primario en SOA	R	31335 (40.54 %)	21070 (27.26 %)	24888 (32.20 %)
Formato Número Serial	R	37202 (48.13 %)	15461 (20.00 %)	24630 (31.87 %)
Tiempos del SOA	R	42283 (54.70 %)	11547 (14.94 %)	23463 (30.36 %)
Tiempo de Refresco	R	7597 (9.83 %)	41045 (53.10 %)	28651 (37.07 %)
Tiempo de Reintento	R	51069 (66.07 %)	2652 (3.43 %)	23572 (30.50 %)
Tiempo de Expiración	R	5442 (7.04 %)	42700 (55.24 %)	29151 (37.71 %)
TTL	R	34079 (44.09 %)	19097 (24.71 %)	24117 (31.20 %)
Registro WWW	R	31310 (40.51 %)	19279 (24.94 %)	26704 (34.55 %)
Registro MX	R	27649 (35.77 %)	5307 (6.87 %)	44337 (57.36 %)
CNAME para MX	R	27625 (35.74 %)	5331 (6.90 %)	44337 (57.36 %)
Número de registros MX	R	5415 (7.01 %)	28279 (36.59 %)	43599 (56.41 %)

La tabla de resultados anterior, está ordenada de acuerdo al tipo de requerimiento analizado, mostrando en primer lugar los requerimientos **necesarios** que debe cumplir un nombre de dominio y luego los requerimientos **recomendados**. El número de evaluaciones de tipo *No Determinada* se debe en la mayoría de los casos a los servidores de nombres que no responden, casos que se diferencian de cuando un servidor de nombres entrega una respuesta errónea a la evaluación. En

algunos casos, este porcentaje es 0 %, lo que se debe a que la evaluación no depende de la respuesta del servidor, por ejemplo el número de servidores de nombres de un nombre de dominio.

Cabe mencionar, que el porcentaje de evaluaciones que no se lograron determinar está entre el 30 % y 40 % en la mayoría de los casos en que se analiza la respuesta del servidor de nombres, indicando que para un mismo grupo de servidores de nombres no se logró determinar la evaluación.

En la sección 8.4 del capítulo de Anexos, se muestran los correspondientes gráficos de todas las evaluaciones realizadas en esta primera prueba.

6.3.2. Datos Relacionados a la Evaluación

Dentro de la evaluación se encontró la siguiente información relevante, con la cual se pueden tomar acciones necesarias para mejorar el estado del DNS.

- **Servidores más populares:** De los datos recopilados de la zona de .CL, se obtiene que los servidores de nombres más populares, junto con el número de nombre de dominios que sirven son:

Número	**Nombre Servidor**
16741	secundario.nic.cl.
4844	polux.entelchile.net.
4332	castor.entelchile.net.
3593	ns1.virtuabyte.com.
3591	ns2.virtuabyte.com.
2591	dns.tie.cl.
2361	ns.puntowebinternet.com.
2317	ns2.puntowebinternet.com.
1740	ns.tchile.com.
1719	ns.attla.cl.
1679	ns2.tchile.com.
1161	ns.wklegal.cl.
1132	ns.gtdinternet.com.
1119	ns2.attla.cl.

En total, este grupo de servidores presta el servicio de DNS a 48920 nombres de dominios, los cuales pueden tener a más de uno de estos servidores registrado en NIC Chile como servidor DNS. Por ejemplo, el nombre de dominio *jose.cl.*, podría tener como servidor DNS primario a: *ns1.virtuabyte.com* y como servidores secundarios a *ns2.virtuabyte.com* y *secundario.nic.cl*. El total de nombres de dominios **diferentes** que atiende este grupo de servidores es: 32435, lo que corresponde al 41.96 % del total de nombres de dominios.

- **Evaluación de Servidores Populares**: Los servidores mencionados en el punto anterior junto con ser los más populares dentro de los nombres de dominio bajo .CL, pasan a ser los más importantes y determinantes en los resultados de esta evaluación, por lo que el mejorar alguna configuración del servicio de DNS que prestan tendrá un gran impacto tanto en el funcionamiento del DNS como en el resultado de la evaluación.

  El resultado de la evaluación del total de las características analizadas en cada uno de los servidores es el siguiente:

Nombre Servidor	**Evaluación**	**Total**	**Porcentaje**
secundario.nic.cl.	Error	87991	22.85 %
	OK	145797	37.87 %
	ND	151255	39.28 %
polux.entelchile.net.	Error	24331	21.84 %
	OK	31277	28.07 %
	ND	55804	50.09 %
castor.entelchile.net.	Error	21811	21.89 %
	OK	27797	27.90 %
	ND	50028	50.21 %
ns1.virtuabyte.com.	Error	26559	32.14 %
	OK	50541	61.16 %
	ND	5539	6.70 %
ns2.virtuabyte.com.	Error	26545	32.14 %
	OK	50505	61.15 %
	ND	5543	6.71 %
dns.tie.cl.	Error	21825	36.62 %
	OK	28366	47.60 %
	ND	9402	15.78 %
ns.puntowebinternet.com.	Error	12690	23.37 %
	OK	30392	55.97 %
	ND	11221	20.66 %
ns2.puntowebinternet.com.	Error	12470	23.40 %
	OK	30102	56.49 %
	ND	10719	20.11 %

ns.tchile.com.	Error	5434	13.58 %
	OK	12402	30.99 %
	ND	22184	55.43 %
ns.attla.cl.	Error	10706	27.08 %
	OK	15600	39.46 %
	ND	13231	33.46 %
ns2.tchile.com.	Error	4989	12.92 %
	OK	11633	30.12 %
	ND	21995	56.96 %
ns.wklegal.cl.	Error	2360	8.84 %
	OK	2342	8.77 %
	ND	22001	82.39 %
ns.gtdinternet.com.	Error	7959	30.57 %
	OK	11759	45.16 %
	ND	6318	24.27 %
ns2.attla.cl.	Error	7694	29.89 %
	OK	10761	41.81 %
	ND	7282	28.29 %

En donde la evaluación 'Error' corresponde a respuestas incorrectas, 'OK' corresponde a respuestas correctas de acuerdo a lo que se está evaluando y 'ND' corresponde a que no se logró determinar la evaluación de acuerdo a la respuesta recibida (o no recibida).

- **Servidores con más nombres**: Dentro de la información que se puede obtener de los datos recopilados en el estudio, están los nombres de servidores DNS que las personas asocian al dominio en NIC Chile. Por ejemplo, para el caso de los segundos servidores más famosos, llamados *polux.entelchile.net.* y *castor.entelchile.net.* se tiene la siguiente variedad de nombres:

```
caftor.entelchile.net.
calox.entelchile.net.
casto.entelchile.net.
castor-entelchile.net.
castor-viejo.entelchile.net.
castor.entelchile.
castor.entelchile.cl.
castor.entelchile.ent.
castor.entelchile.net.
castorentelchile.net.
castos.entelchile.net.
castpr.entelchile.net.
```

```
castro.entelchile.net.
castror.entelchile.net.
costos.entelchile.net.
custom.entelchile.net.
plux.entelchile.net.
polix.entelchile.net.
polloux.entelchile.net.
pollux.entelchile.net.
polox.entelchile.net.
polux-back.entelchile.net.
polux-entelchile.net.
polux-viejo.entelchile.net.
polux.entelchile.
polux.entelchile.cl.
polux.entelchile.net.
poluxentelchile.net.
poluxviejo.entelchile.net.
poluxx.entelchile.net.
polx.entelchile.net.
ponux.entelchile.net.
```

Dentro de este listado, claramente hay nombres de servidores que no corresponden a la realidad, y estos datos son originados por los clientes de NIC Chile, ya que son ellos los que escriben el nombre de los servidores DNS del nombre de dominio en cuestión. Claramente, todas las evaluaciones realizadas para un servidor con nombre erróneo no serán exitosas, afectando el resultado general de la evaluación.

De los datos relacionados con la evaluación, mostrados en los ítemes anteriores, se puede tener una visión global de lo que está sucediendo con los servidores DNS de los nombres de dominios .CL. Luego de analizar estos datos, es necesario tomar las acciones pertinentes para mejorar el estado del DNS en .CL, tomando como partida el nombre de estos servidores famosos, los que de mejorar sus respuestas a las evaluaciones realizadas, mejorarán a su vez el resultado total de la evaluación.

6.4. Medidas Recomendadas

Las medidas que se recomiendan seguir, de acuerdo a los resultados de la evaluación realizada son:

- De acuerdo a las 2 características peor evaluadas, referentes a 'Servidores Activos' con 65.45 % de error (de tipo necesaria) y 'Completitud de Servidores' con 75.88 % de error (de tipo recomendada), se recomienda encarecidamente sugerir a los clientes de NIC Chile y administradores de los servidores DNS mencionados en la sección 6.3.2 de este capítulo, que se revise que los servidores de DNS del(los) nombre(s) de dominio(s) que tengan a su cargo estén respondiendo por éste y además, dentro de esa respuesta incluyan a los otros servidores de nombre registrados en NIC Chile. De no estar correcta la información registrada en NIC Chile, se debería modificar para que figuren los datos correctos.

- Otra de las característica necesarias peor evaluada es 'Reversos de Servidores', con un 55.32 % de error. Se recomienda generar una publicación recordando a los administradores de los servidores DNS la importancia de tener bien configurado los reversos, el problema principal es que la mayoría no toma la debida importancia hasta que se encuentra con serios problemas de funcionalidad de las aplicaciones que antes no tenía.

- Las características con mayor porcentaje de resultados no determinados, son las relacionadas al registro de tipo MX, considerando que sólo para un 35.77 % de los nombres de dominio evaluados se logró determinar la existencia de este registro y que sólo el 7.01 % de los nombres de dominios tiene más de 1 registro MX, se recomienda generar alguna publicación que muestre la importancia de tener más de un registro MX por nombre de dominio.

- Las características recomendadas de 'No Recursividad' y 'Versiones de Software DNS', tienen un alto porcentaje de error: 49.83 % y 62.61 %. Se recomienda generar alguna publicación en la que se mencione los posibles problemas de seguridad a los que se puede ver afectado un nombre de dominio que tiene servidores DNS que responden de manera recursiva y que además tiene las mismas versiones de software DNS funcionando en sus servidores.

- La característica mejor evaluada es la del 'Número de Servidores', en donde tiene gran impacto la existencia del servidor DNS *secundario.nic.cl*, el cual al prestar el servicio de forma gratuita y de fácil acceso desde la inscripción del nombre de dominio es bastante utilizado por los clientes de NIC Chile.

- Dada la gran cantidad de nombres erróneos que puede tener un servidor DNS, como se mostró en la subsección 6.3.2 de este capítulo, se recomienda alertar a los clientes de NIC Chile para que revisen los datos relacionados a los nombres de los servidores DNS asociados

a su nombre de dominio. Por otro lado, se recomienda a NIC Chile realizar una mejor revisión de nombre de servidor DNS incluido por el cliente, ya sea al momento de la inscripción del nombre de dominio o al momento de la incorporación de éste a la zona de .CL.

Dado que este estudio es el primero que se realiza en .CL, este grupo de recomendaciones sólo es inicial y podría crecer mucho más en base a la opinión y conversaciones con el grupo de Ingeniería de NIC Chile.

Capítulo 7

Conclusiones

El diseño del sistema de nombres de dominio, ha demostrado ser lo suficientemente robusto al soportar sin mayores problemas el explosivo crecimiento de Internet. Por otro lado, los oportunos RFCs publicados han permitido definir recomendaciones generales para que este sistema funcione lo mejor posible. Sin embargo, las personas que administran servidores DNS no han demostrado cumplir estas recomendaciones de la mejor manera, lo que se ve reflejado en el resultado de la prueba realizada con esta tesis en los dominios .CL.

La evaluación del estado del DNS, sirve para conocer qué tanto se están cumpliendo las recomendaciones internacionales para este sistema. También, sirve para detectar algún problema dentro del funcionamiento de los servidores DNS y de paso prevenir su crecimiento. Con la publicación del resultado de la evaluación, se está educando a los administradores del servicio de DNS, que ahora tendrán un lugar en donde podrán conocer en que estado están los servidores y nombres de dominio que están a su cargo. A nivel internacional, la preocupación por conocer el estado del DNS ha crecido con el tiempo, alcanzando un mínimo nivel de importancia que este sistema merece. Esto se debe en gran medida a los últimos ataques realizados a los servidores raíz, intentando hacer una denegación de servicios.

El desarrollo de una herramienta para evaluar el estado del DNS es una tarea bastante complicada, comenzando por definir las características que se desean evaluar, para luego llevarlas a la implementación. Además, si se agrega que la herramienta se probará para un TLD con aproximadamente 75000 nombres de dominios aparece la necesidad de implementar mecanismos de

cache interno, con el fin de optimizar el tiempo de ejecución.

De acuerdo al resultado de la prueba realizada para .CL, se tiene que la gran fuente de errores dentro del sistema DNS es el descuido de los administradores y clientes de NIC Chile, los que no ponen especial cuidado en asociar como corresponde un servidor DNS a un nombre de dominio, cometiendo errores en escribir el nombre del servidor, asociando servidores que no corresponden al nombre de dominio, y asociando servidores que simplemente tienen su nombre mal formado. Otra gran fuente de errores, de parte de los administradores de servidores, es el no hacer caso a las recomendaciones internacionales como nacionales, sobre temas de seguridad en los servidores DNS, manteniendo configuraciones vulnerables a un ataque.

De las 6 características necesarias que debe cumplir un nombre de dominio, se tiene que para .CL se cumplen en un 44.62 %, lo cual muestra un estado regular del DNS. El mayor problema proviene de las configuraciones erróneas de los servidores, ya que se tiene un 65.45 % de servidores que están activos y respondiendo para los dominios, pero las respuestas que se obtienen demuestran problemas y errores en la configuración.

La explosiva masificación de Internet va de la mano con el aumento de los nombres de dominio. Cada día se crean nuevos negocios, comunicaciones y servicios utilizando esta red, generando una gran dependencia de un servicio que puede verse interrumpido si el sistema de DNS comienza a presentar errores. Es necesario dar la importancia que merece al DNS, enseñarlo y fomentar las recomendaciones internacionales en el medio local, para que en un futuro no se lamenten los errores que se alertan hoy en día.

7.1. Trabajo Futuro

El detectar los problemas, clasificarlos y publicarlos es sólo el primer paso para mejorar el estado del DNS. Ahora queda la importante tarea de difundir los reportes y realizar las comunicaciones necesarias con los administradores de los servidores de nombres más famosos dentro de .CL para sugerir las recomendaciones necesarias y así solucionar los problemas detectados con este estudio.

Dentro de los resultados obtenidos para .CL existe un alto porcentaje de características con evaluación no determinada, en la mayoría de los casos se debe a que no se logró una comunicación

con el servidor de nombres, el cual se almacena en un cache interno que mantiene los nombres de los servidores que no están alcanzables. Como la herramienta demoró alrededor de 40 horas en la ejecución, perfectamente un servidor podría volver a estar alcanzable, por lo que sería necesario incorporar un mecanismo de cache dinámico, el cual cumplido cierto tiempo vuelve a consultar al servidor marcado como inalcanzable. De esta manera, podría disminuir la cantidad de evaluaciones no determinadas.

Con el paso del tiempo, es probable que aparezcan nuevas recomendaciones internacionales para el DNS. Es necesario adaptar la herramienta implementada a las nuevas necesidades, como mejorar las posibles fallas que puede tener la implementación actual. Es necesario mejorar la presentación de los reportes, esa importante información debe ser presentada en un formato amigable para que los usuarios puedan recibir el mensaje de alerta e informativo que se pretende entregar, en donde el autor de esta tesis no puede colaborar en demasía por no tener los conocimientos de diseño necesario.

Es necesario seguir aplicando este estudio, siendo ideal generar reportes de manera periódica que puedan ser comparados con reportes anteriores y permitan ver claramente las mejoras y nuevas fallas que se encuentran. Como la distribución de la herramienta será de manera gratuita, se espera generar nuevas versiones y mejoras del código fuente de toda la herramienta. También, se pueden agregar nuevos módulos que permitan generar nuevos reportes automáticos a partir de la información recopilada durante el estudio.

Un punto de comparación bastante bueno sería el aplicar este mismo estudio a otro TLD, de paso serviría para probar la herramienta en otro ambiente y lo más importante, para comparar el estado del DNS entre distintos TLDs medidos con la misma herramienta.

Capítulo 8

Anexos

8.1. Definiciones

- *browser*: Software que permite visualizar los contenidos de los sitios web disponibles en Internet. Los más conocidos son: Internet Explorer, Netscape, Opera, Mozilla.
- *dirección IP*: Identificador único de un computador conectado a Internet, usado para que los computadores puedan comunicarse entre sí. Por ejemplo, para el nombre uchile.cl el número IP asociado es: 146.83.12.3.
- *ICANN*: Internet Corporation for Assigned Names and Numbers, Institución encargada de la administración del Sistema de Nombres de Dominios en Internet. Más información en: *http://www.icann.org*
- *RIPE*: Réseaux IP Européens, es un foro colaborativo abierto a todas las partes interesadas en el amplia area de redes IP. Su objetivo es asegurar la coordinación técnica y administrativa necesaria para facilitar la operación de Internet en la región de RIPE. Más información en *http://www.ripe.net/*.
- *IETF*: The Internet Engineering Task Force, es una gran comunidad de diseñadores de redes, operadores e investigadores preocupados de la evolución de la arquitectura de Internet. Más información en: *http://www.ietf.org/*
- *ccTLDs*: (Country Codes Top Level Domain) Corresponden a los nombres de dominio de

primer nivel que identifican países, por ejemplo .CL para Chile, .BR para Brasil, .MX para México, etcétera.

- *UDP*: User Datagram Protocol, es un protocolo de comunicaciones que ofrece una cantidad limitada de servicios para las transmisiones entre computadores que usan el Internet Protocol (IP). Es una alternativa al Transmission Control Protocol (TCP), su ventaja está en que no crea conexiones para realizar las transmisiones y su desventaja es que no hay control de la transmisión misma.
- *TCP*: Transmission Control Protocol, es un protocolo con un conjunto de reglas usado sobre Internet Protocol (IP), para enviar datos en forma de unidades de mensajes entre computadores sobre Internet. Mientras IP se preocupa de manejar la entrega de los datos, TCP se preocupa de guardar un rastro de cada unidad de datos (llamados paquetes) del mensaje, el cual es dividido para un ruteo más eficiente sobre Internet.
- *web hosting*: Servicio provisto generalmente por proveedores de Internet (ISP), mediante el cual se puede mantener los archivos de uno o más sitios web en un servidor.
- *mail hosting*: Servicio que realiza la mantención y administración de los emails para algún dominio.
- *dig*: Herramienta que es utilizada para realizar diversas consultas a los servidores DNS.
- *CGI*: Es la vía estándar por medio de la cual un servidor Web entrega los requerimientos de un usuario a una aplicación y recibe la respuesta de la aplicación para entregársela al usuario.

8.2. ccTLDs

TLD	País
.ac	Ascension Island
.ad	Andorra
.ae	United Arab Emirates
.af	Afghanistan
.ag	Antigua and Barbuda
.ai	Anguilla
.al	Albania
.am	Armenia
.an	Netherlands Antilles
.ao	Angola
.aq	Antarctica
.ar	Argentina
.as	American Samoa
.at	Austria
.au	Australia
.aw	Aruba
.az	Azerbaijan
.ba	Bosnia and Herzegovina
.bb	Barbados
.bd	Bangladesh
.be	Belgium
.bf	Burkina Faso
.bg	Bulgaria
.bh	Bahrain
.bi	Burundi
.bj	Benin
.bm	Bermuda
.bn	Brunei Darussalam
.bo	Bolivia
.br	Brazil
.bs	Bahamas
.bt	Bhutan
.bv	Bouvet Island
.bw	Botswana
.by	Belarus
.bz	Belize

TLD	País
.ca	Canada
.cc	Cocos (Keeling) Islands
.cd	Congo, Democratic Republic of the
.cf	Central African Republic
.cg	Congo, Republic of
.ch	Switzerland
.ci	Cote d'Ivoire
.ck	Cook Islands
.cl	Chile
.cm	Cameroon
.cn	China
.co	Colombia
.cr	Costa Rica
.cu	Cuba
.cv	Cap Verde
.cx	Christmas Island
.cy	Cyprus
.cz	Czech Republic
.de	Germany
.dj	Djibouti
.dk	Denmark
.dm	Dominica
.do	Dominican Republic
.dz	Algeria
.ec	Ecuador
.ee	Estonia
.eg	Egypt
.eh	Western Sahara
.er	Eritrea
.es	Spain
.et	Ethiopia
.fi	Finland
.fj	Fiji
.fk	Falkland Islands (Malvina)
.fm	Micronesia, Federal State of
.fo	Faroe Islands
.fr	France
.ga	Gabon
.gd	Grenada
.ge	Georgia

TLD	País
.gf	French Guiana
.gg	Guernsey
.gh	Ghana
.gi	Gibraltar
.gl	Greenland
.gm	Gambia
.gn	Guinea
.gp	Guadeloupe
.gq	Equatorial Guinea
.gr	Greece
.gs	South Georgia and the South Sandwich Islands
.gt	Guatemala
.gu	Guam
.gw	Guinea-Bissau
.gy	Guyana
.hk	Hong Kong
.hm	Heard and McDonald Islands
.hn	Honduras
.hr	Croatia/Hrvatska
.ht	Haiti
.hu	Hungary
.id	Indonesia
.ie	Ireland
.il	Israel
.im	Isle of Man
.in	India
.io	British Indian Ocean Territory
.iq	Iraq
.ir	Iran (Islamic Republic of)
.is	Iceland
.it	Italy
.je	Jersey
.jm	Jamaica
.jo	Jordan
.jp	Japan
.ke	Kenya
.kg	Kyrgyzstan
.kh	Cambodia
.ki	Kiribati
.km	Comoros

TLD	País
.kn	Saint Kitts and Nevis
.kp	Korea, Democratic People’s Republic
.kr	Korea, Republic of
.kw	Kuwait
.ky	Cayman Islands
.kz	Kazakhstan
.la	Lao People’s Democratic Republic
.lb	Lebanon
.lc	Saint Lucia
.li	Liechtenstein
.lk	Sri Lanka
.lr	Liberia
.ls	Lesotho
.lt	Lithuania
.lu	Luxembourg
.lv	Latvia
.ly	Libyan Arab Jamahiriya
.ma	Morocco
.mc	Monaco
.md	Moldova, Republic of
.mg	Madagascar
.mh	Marshall Islands
.mk	Macedonia, Former Yugoslav Republic
.ml	Mali
.mm	Myanmar
.mn	Mongolia
.mo	Macau
.mp	Northern Mariana Islands
.mq	Martinique
.mr	Mauritania
.ms	Montserrat
.mt	Malta
.mu	Mauritius
.mv	Maldives
.mw	Malawi
.mx	Mexico
.my	Malaysia
.mz	Mozambique
.na	Namibia
.nc	New Caledonia

TLD	País
.ne	Niger
.nf	Norfolk Island
.ng	Nigeria
.ni	Nicaragua
.nl	Netherlands
.no	Norway
.np	Nepal
.nr	Nauru
.nu	Niue
.nz	New Zealand
.om	Oman
.pa	Panama
.pe	Peru
.pf	French Polynesia
.pg	Papua New Guinea
.ph	Philippines
.pk	Pakistan
.pl	Poland
.pm	St. Pierre and Miquelon
.pn	Pitcairn Island
.pr	Puerto Rico
.ps	Palestinian Territories
.pt	Portugal
.pw	Palau
.py	Paraguay
.qa	Qatar
.re	Reunion Island
.ro	Romania
.ru	Russian Federation
.rw	Rwanda
.sa	Saudi Arabia
.sb	Solomon Islands
.sc	Seychelles
.sd	Sudan
.se	Sweden
.sg	Singapore
.sh	St. Helena
.si	Slovenia
.sj	Svalbard and Jan Mayen Islands
.sk	Slovak Republic

TLD	País
.sl	Sierra Leone
.sm	San Marino
.sn	Senegal
.so	Somalia
.sr	Suriname
.st	Sao Tome and Principe
.sv	El Salvador
.sy	Syrian Arab Republic
.sz	Swaziland
.tc	Turks and Caicos Islands
.td	Chad
.tf	French Southern Territories
.tg	Togo
.th	Thailand
.tj	Tajikistan
.tk	Tokelau
.tm	Turkmenistan
.tn	Tunisia
.to	Tonga
.tp	East Timor
.tr	Turkey
.tt	Trinidad and Tobago
.tv	Tuvalu
.tw	Taiwan
.tz	Tanzania
.ua	Ukraine
.ug	Uganda
.uk	United Kingdom
.um	US Minor Outlying Islands
.us	United States
.uy	Uruguay
.uz	Uzbekistan
.va	Holy See (City Vatican State)
.vc	Saint Vincent and the Grenadines
.ve	Venezuela
.vg	Virgin Islands (British)
.vi	Virgin Islands (USA)
.vn	Vietnam
.vu	Vanuatu
.wf	Wallis and Futuna Islands

TLD	País
.ws	Western Samoa
.ye	Yemen
.yt	Mayotte
.yu	Yugoslavia
.za	South Africa
.zm	Zambia
.zw	Zimbabwe

8.3. Manual de Usuario

Bienvenido al sistema de evaluación del estado del DNS **'Inspector DNS'**. En este manual de usuario encontrará todo lo necesario para conocer el sistema, configurarlo, instalarlo, ejecutarlo, generar reportes y solucionar los problemas que se puedan presentar.

8.3.1. Acerca del Sistema

Este sistema fue desarrollado en su primera versión como parte de la Tesis de Magíster del autor de este manual, durante el año 2003. Esta primera versión, se aplicó para evaluar el estado del DNS de los dominios inscritos bajo el TLD .CL.

8.3.2. Requisitos

Para que el sistema opere correctamente es necesario que el computador bajo el cual operará tenga el siguiente software instalado:

- Motor de Base de Datos MySql, versión 4.0.16 o superior.

- Lenguaje Perl, versión 5.8.0 o superior.

El lenguaje Perl debe ser complementado con los módulos DBI, GD, Net::DNS y DNS::ZoneParse. Estos módulos pueden ser descargados desde el sitio: *http://www.cpan.org*.

8.3.3. Instalación y Configuración

Una vez descargado el sistema desde el sitio web de NIC Chile[1] lo debe descomprimir y dar permisos de ejecución a los programas ejecutables: inspector.pl, reporte.pl, reporteDetalle.pl. Además, se deben dar permisos de lectura y ejecución a las bibliotecas de funciones y configuración: LibInspector.pm, ConfigInspector.pm.

Los parámetros de configuración del sistema se manejan en el archivo llamado *ConfigInspector.pm*, en el cual se deben asignar los valores correctos a las siguientes variables:

- **dbname**: Corresponde al nombre de la base de datos.
- **dbhost**: El nombre del computador que será el servidor de la base de datos.
- **dbuser**: El nombre de usuario de la base de datos.
- **dbpass**: La clave correspondiente al nombre de usuario, la cual se utilizará para conectarse a la base de datos.
- **fileZone**: Archivo correspondiente a la zona del TLD que se quiere analizar.
- **fileSQL**: Archivo correspondiente a las sentencias SQL de la creación de la base de datos.
- **fileCarac**: Archivo que posee el nombre y descripción de las características que analiza la herramienta.
- **TLD**: Sufijo para el cual se está realizando la evaluación. Para el caso de NIC Chile, tiene como valor '.cl'.

[1] http://www.nic.cl

El archivo de la zona que se desea analizar, debe estar en la ruta de archivo mencionada en la variable **fileZone**, y por lo menos debe tener permisos de lectura.

Base de datos

Luego de asignar los valores correspondientes a las variables del sistema, es necesario crear la base de datos que almacenará los resultados de la evaluación. Dentro del archivo de configuración del sistema (ConfigInspector.pm) se sugiere que el nombre de la base de datos sea **INSPECTORDNSAAAA_MM**, en donde 'AAAA' corresponde al año actual y 'MM' al mes. Estos valores se obtienen de la fecha del servidor en donde se ejecuta la herramienta. De más está mencionar que estos valores los puede modificar y usar el nombre de base de datos que usted desee.

Una vez creada la base de datos, es necesario dar los permisos necesarios para el nombre de usuario asignado a la variable **dbuser** en el archivo de configuración, el cual se identificará por medio de la clave asignada en la variable **dbpass**. Los permisos deben ser de creación, actualización e inserción en la base de datos.

8.3.4. Ejecución

Luego de haber asignado las variables necesarias en el archivo de configuración, asignar los permisos necesarios al usuario en la base de datos y tener el archivo de zona en donde corresponde, se debe ejecutar el programa identificado por el archivo **inspector.pl**.

Durante la ejecución, se irán imprimiendo a la salida estándar mensajes mientras avanza la evaluación, y de paso se insertarán en la base de datos los registros necesarios. En primer lugar se crearán las tablas que componen el modelo de datos, luego se insertarán las descripciones de las características que se están analizando, para finalmente insertar los nombres de dominios con sus correspondientes servidores de nombres asociados.

Cuando se tiene la base de datos poblada con las características, nombres de dominios y servidores de nombres, comenzará la evaluación del DNS, en donde la herramienta recorrerá secuen-

cialmente el listado de dominios, obtendrá el listado de servidores de nombres asociados a esos dominios y aplicará un grupo de consultas DNS a dichos servidores, para posteriormente analizar las respuestas obtenidas y asignar el resultado de la evaluación y las glosas correspondientes.

8.3.5. Generación de Reportes

Una vez que el **inspector.pl** ha terminado su ejecución, se debe seguir con la generación de los reportes, para obtener una visualización detallada del resultado de la evaluación. Para esto, debe ejecutar el programa identificado por el nombre **reporte.pl**, el cual generará dentro del directorio en ejecución un nuevo directorio llamado 'html', en donde se dejará un archivo con extensión '.html' para cada evaluación realizada.

Para complementar la información de los archivos HTML, se creará un directorio llamado 'html/imagenes/', dentro del cual se almacenará un archivo que contendrá una imagen por cada evaluación realizada, la cual es incluida dentro del archivo HTML correspondiente.

Los reportes de cada evaluación mostrarán en primer lugar un gráfico de torta con el resultado, y luego se agregará una descripción de la evaluación, en donde se explica que cosa se evaluó y cuales son las consecuencias de que la evaluación sea negativa.

Reporte por Nombre de Dominio

Los reportes descritos en los párrafos anteriores corresponden al resultado global de la evaluación, mostrando sólo los porcentajes de evaluaciones exitosas y evaluaciones erróneas. Pero también existe la manera de saber como resultó la evaluación por cada nombre de dominio, esto es mediante la ejecución del CGI **reporteDetalle.pl**, el cual solicita un nombre de dominio y muestra el resultado de la evaluación de él.

En el caso que los reportes detallados se deseen mostrar por medio de un sitio web, es necesario configurar la ejecución de CGI escritos en el lenguaje Perl, y en particular, dar la configuración necesaria para que el CGI **reporteDetalle.pl** se ejecute correctamente. Este CGI utiliza el archivo

de configuración de la herramienta, para obtener los valores necesarios de conexión a la base de datos y las sentencias SQL necesarias para obtener el reporte detallado del nombre de dominio en cuestión.

8.3.6. Problemas Comunes

8.4. Gráficos Prueba Realizada

A continuación, se anexan los gráficos asociados a la prueba realizada y descrita en el capitulo 6 de Pruebas. Cada gráfico representa el resultado de evaluar una característica en particular, en los que se aprecian tres colores, para las tres cifras que se midieron: Evaluación correcta, Evaluación Errónea y Evaluación no determinada. Algunos, no traen graficada las cifras para las evaluaciones no determinadas, ya que corresponden a evaluaciones en donde todas las características se evaluaron correctamente o erróneamente.

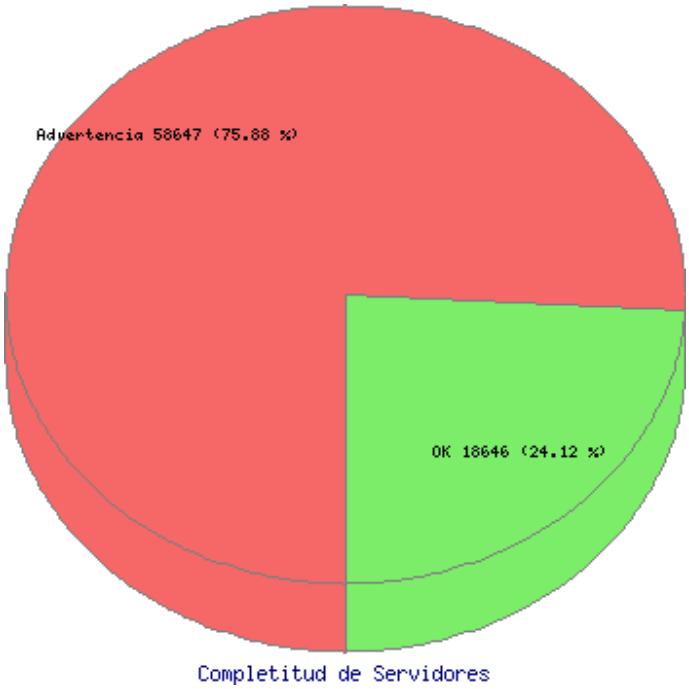

Figura 8.1: Evaluación Completitud de Servidores

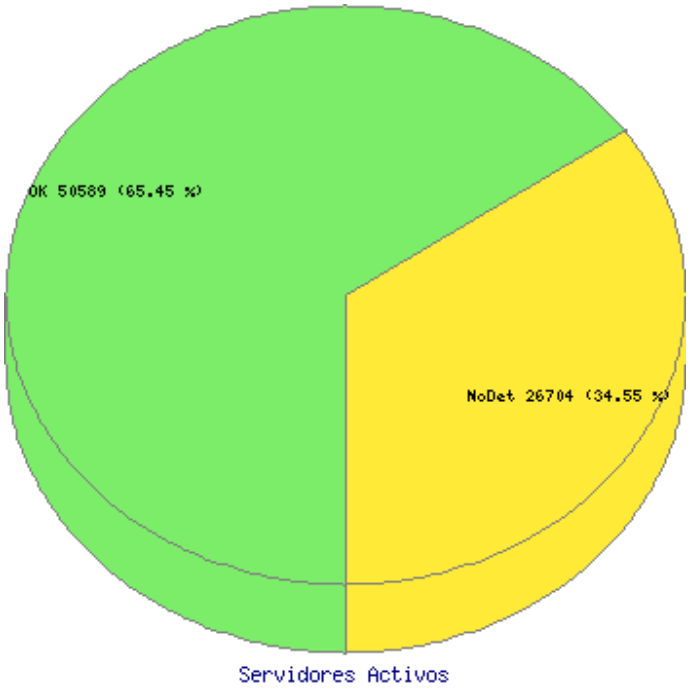

Figura 8.2: Evaluación Servidores Activos

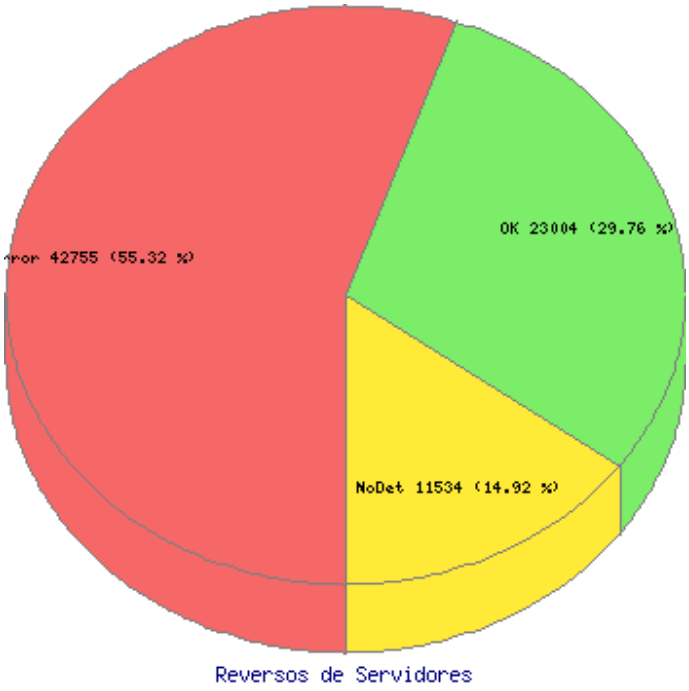

Figura 8.3: Evaluación Reversos de Servidores

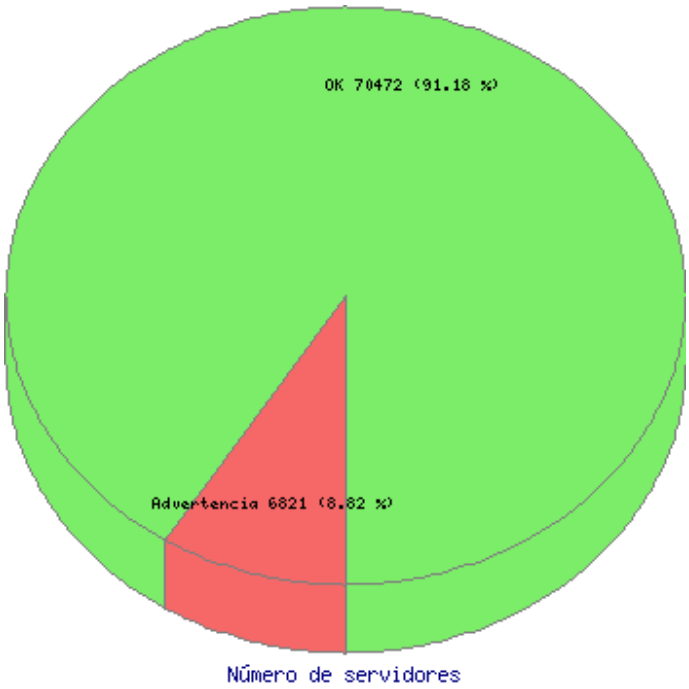

Figura 8.4: Evaluación Número de servidores

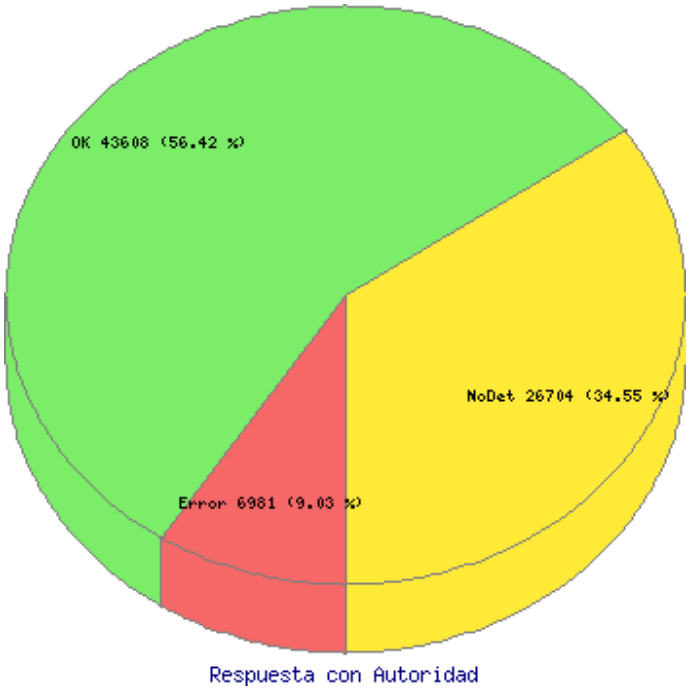

Figura 8.5: Evaluación Respuesta con Autoridad

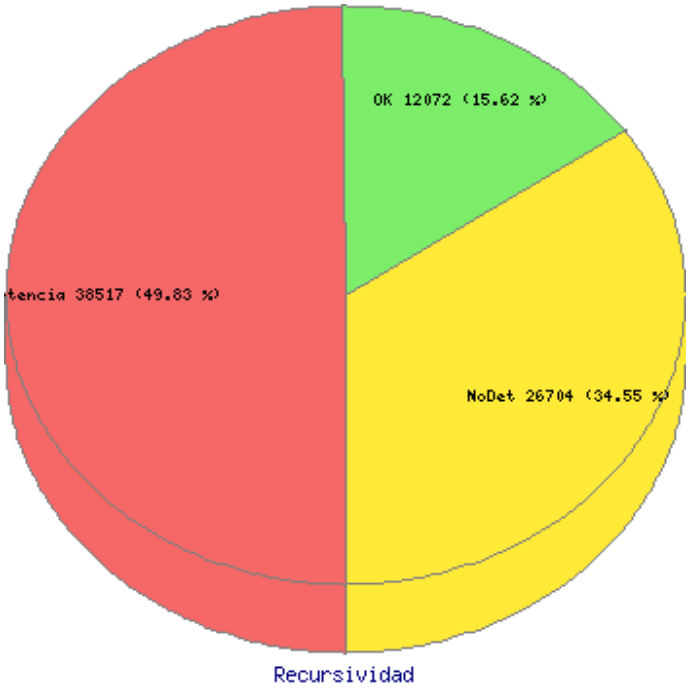

Figura 8.6: Evaluación de Respuestas sin Recursividad

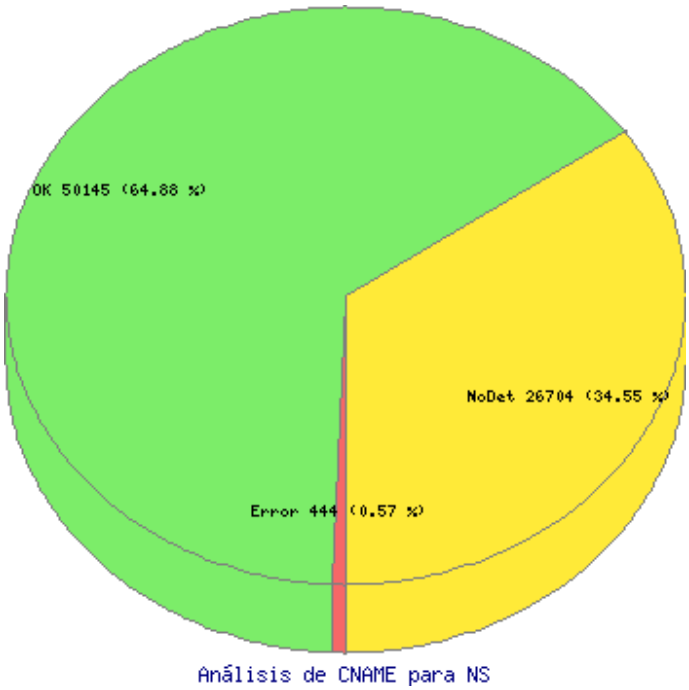

Figura 8.7: Evaluación Análisis de CNAME para NS

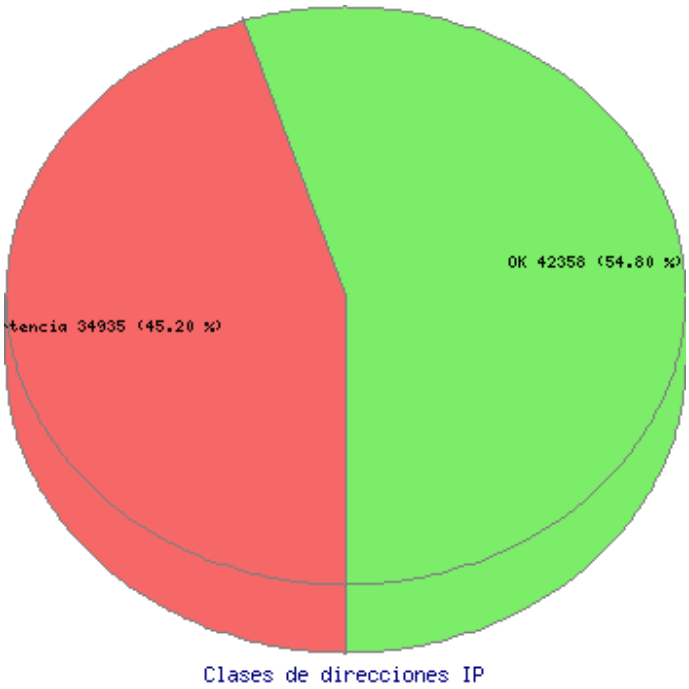

Figura 8.8: Evaluación Clases de direcciones IP

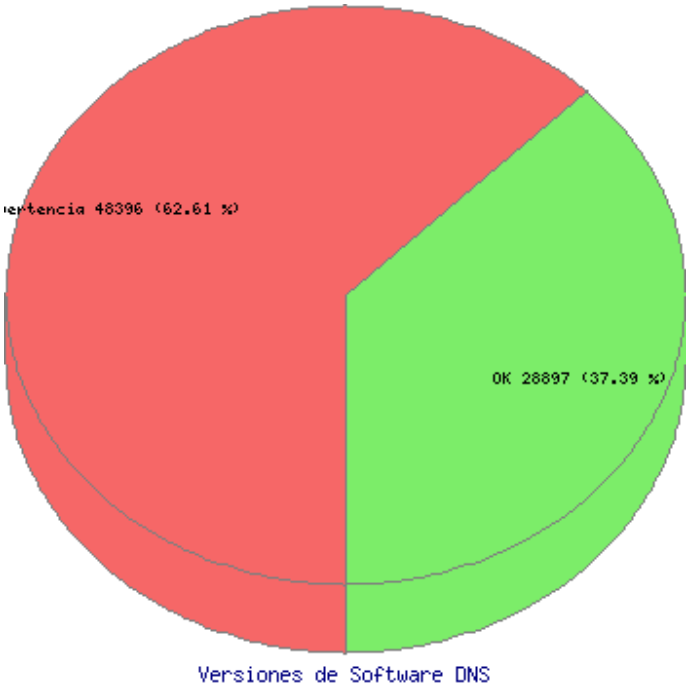

Figura 8.9: Evaluación Versiones de Software DNS

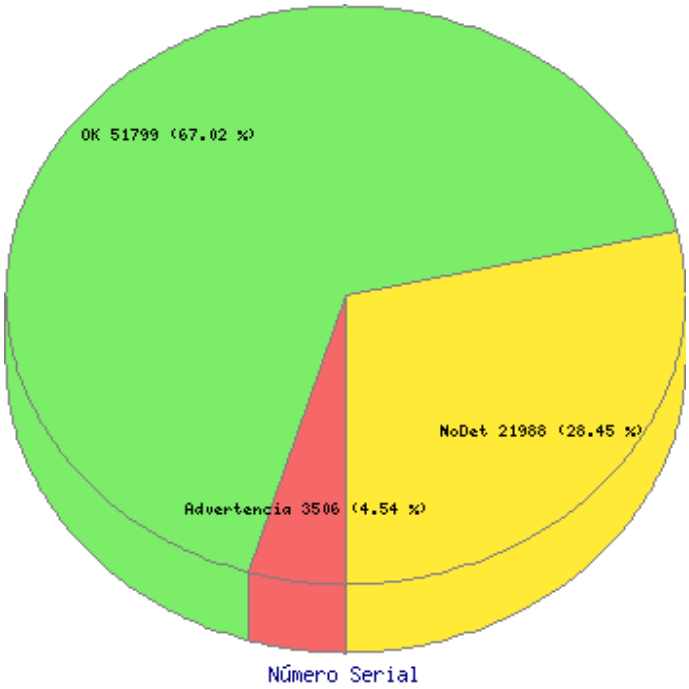

Figura 8.10: Evaluación Número Serial

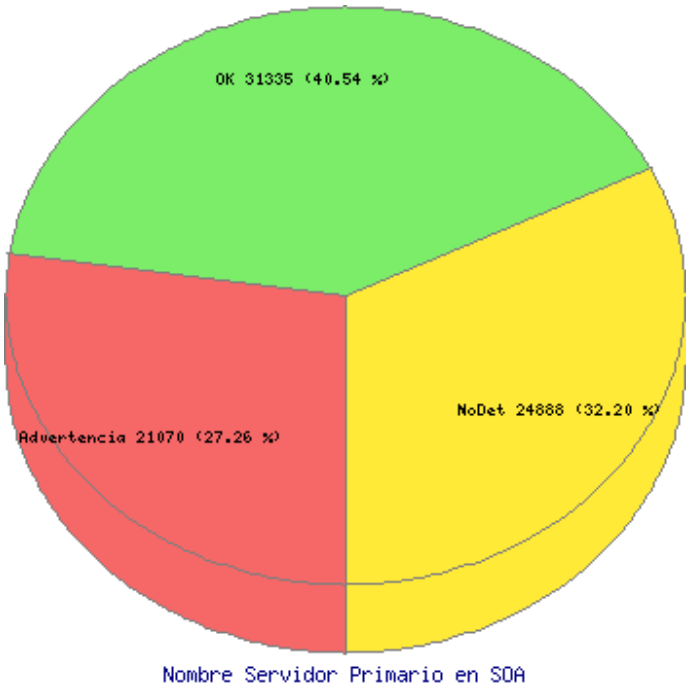

Figura 8.11: Evaluación Nombre Servidor Primario en SOA

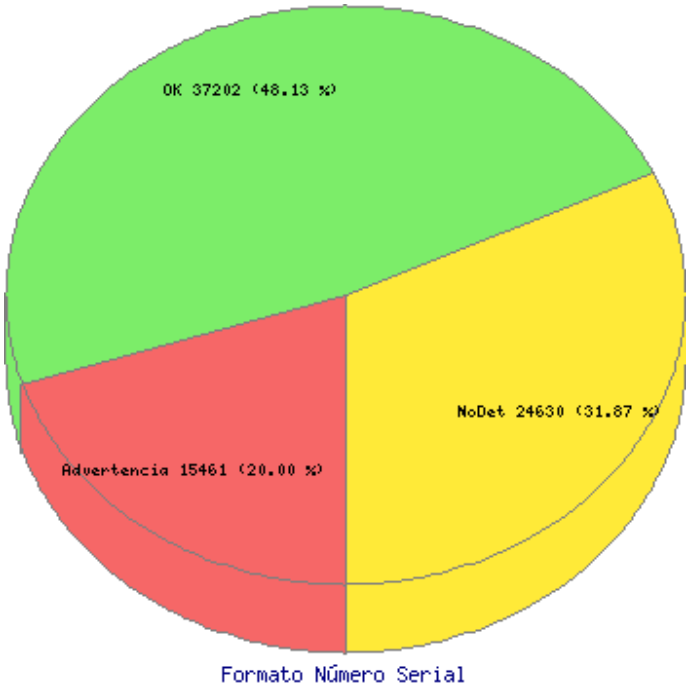

Figura 8.12: Evaluación Formato Número Serial

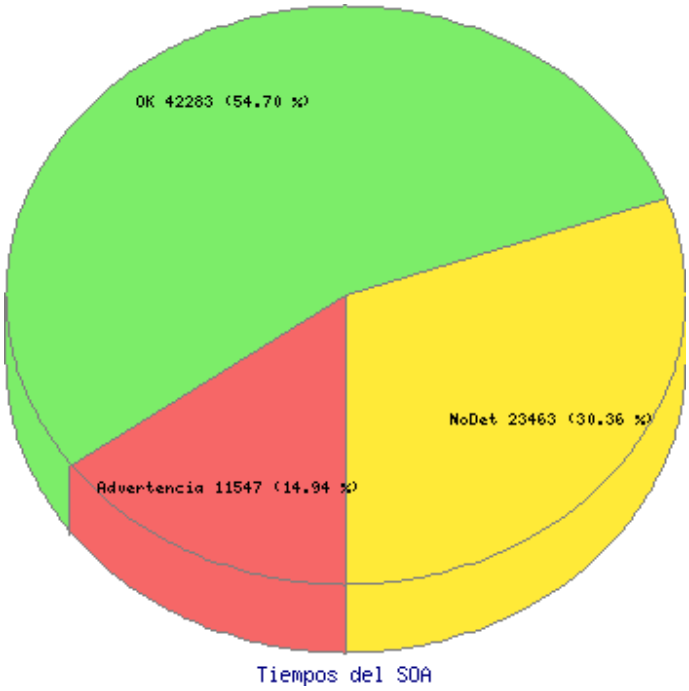

Figura 8.13: Evaluación Tiempos del Registro SOA

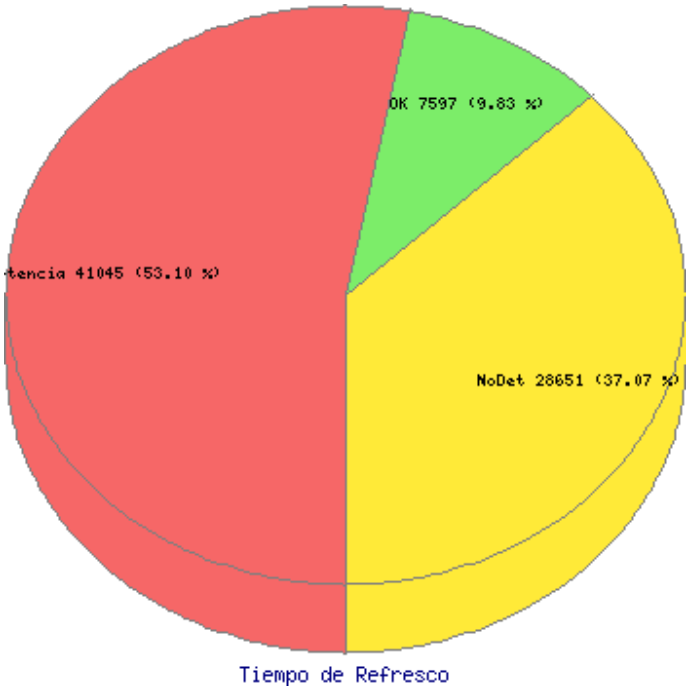

Figura 8.14: Evaluación Tiempo de Refresco

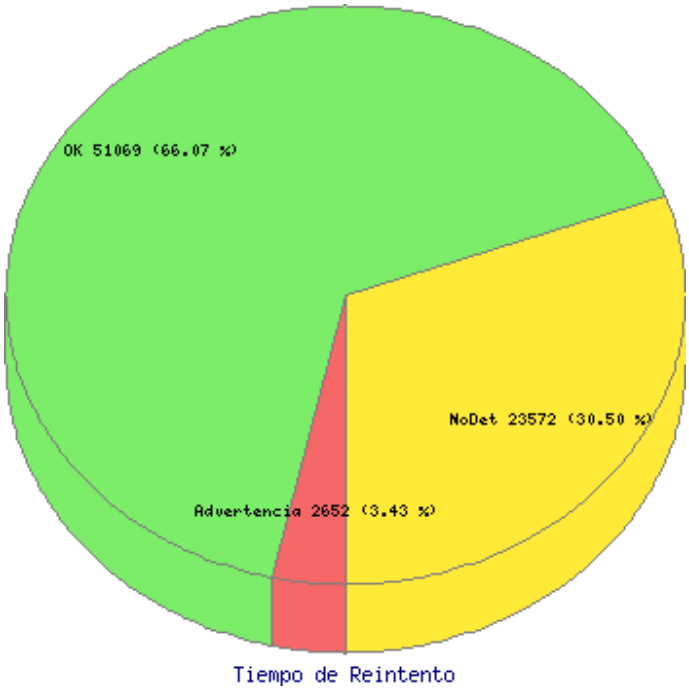

Figura 8.15: Evaluación Tiempo de Reintento

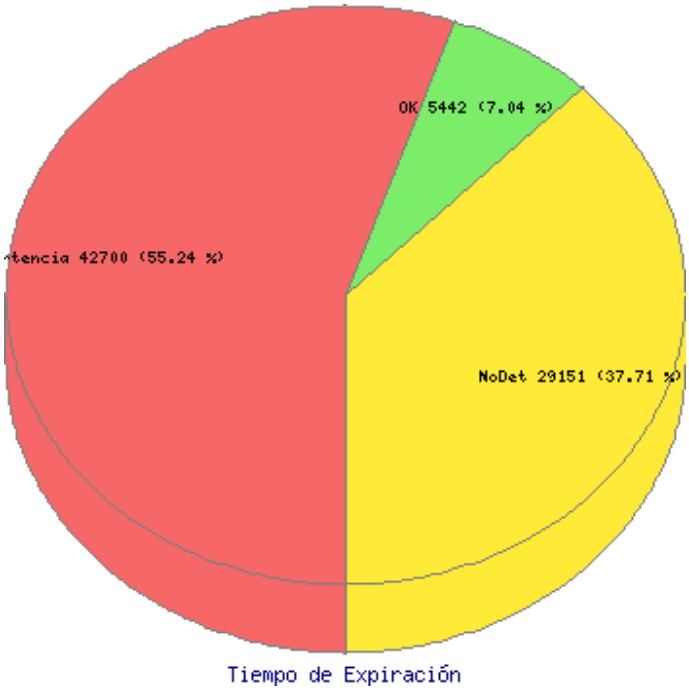

Figura 8.16: Evaluación Tiempo de Expiración

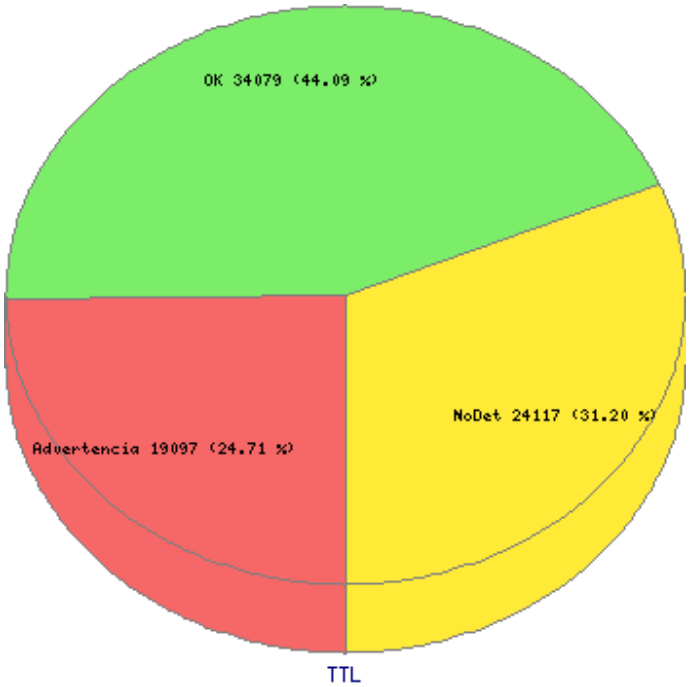

Figura 8.17: Evaluación Tiempo de Time To Live

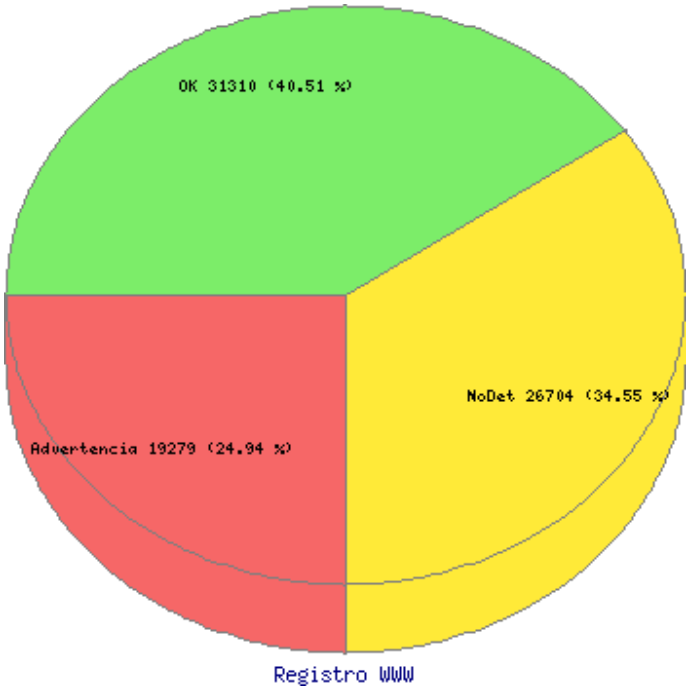

Figura 8.18: Evaluación Existencia de Registro WWW

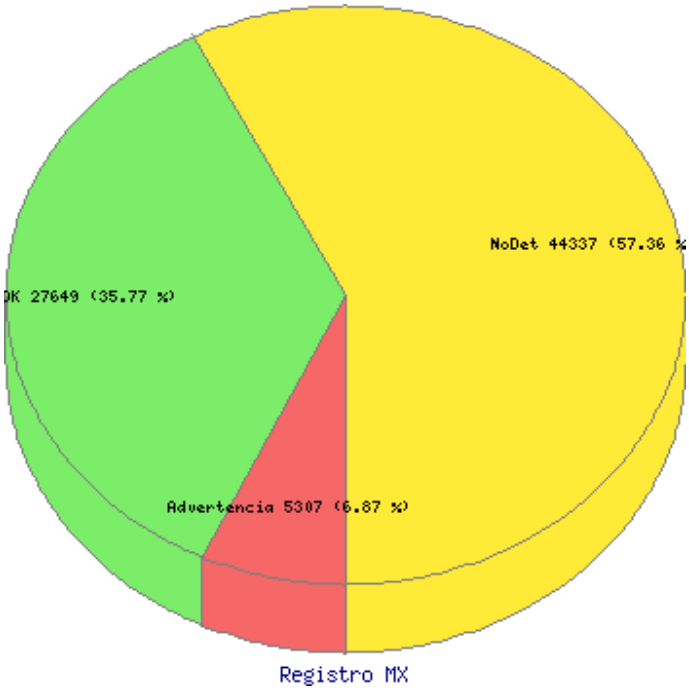

Figura 8.19: Evaluación Existencia de Registro MX

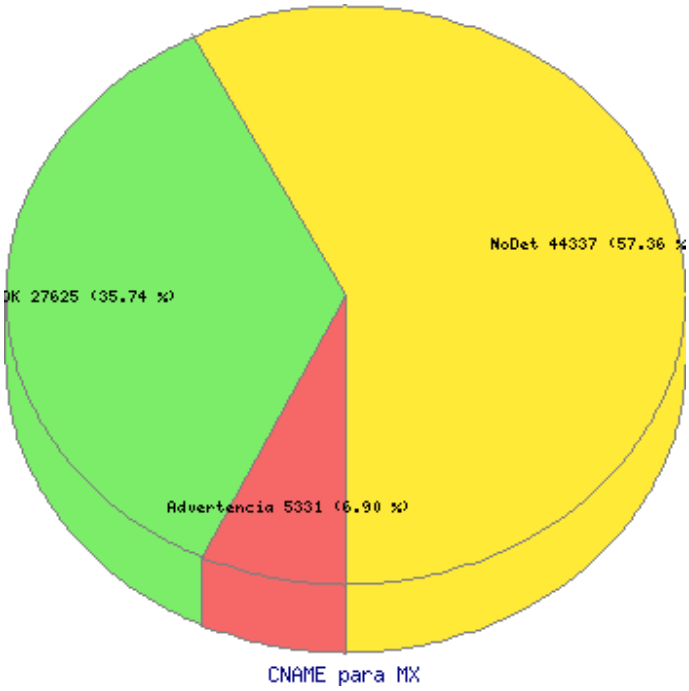

Figura 8.20: Evaluación el Registro MX no debe ser CNAME

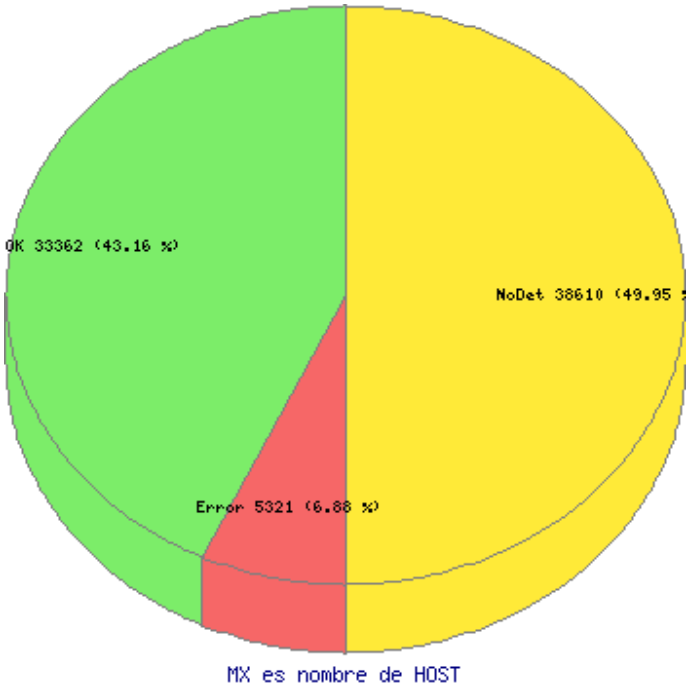

Figura 8.21: Evaluación del Registro MX es nombre de HOST

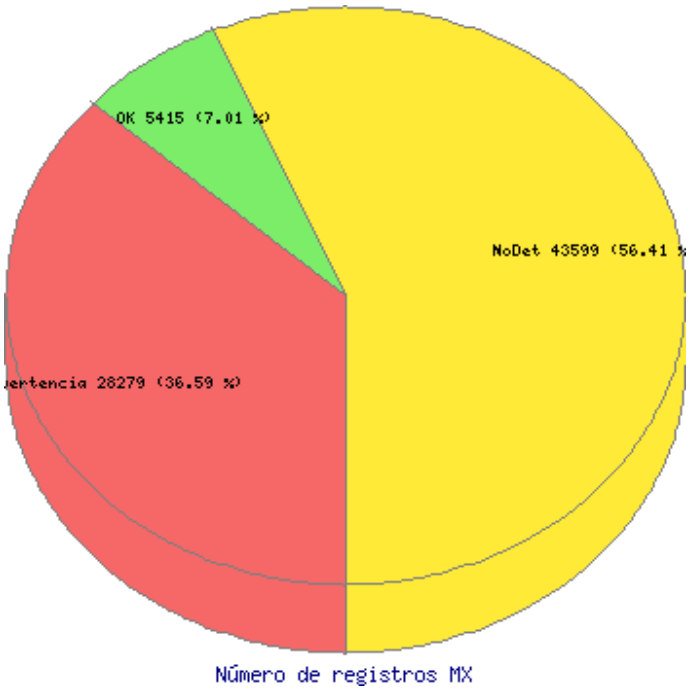

Figura 8.22: Evaluación de Número de registros MX

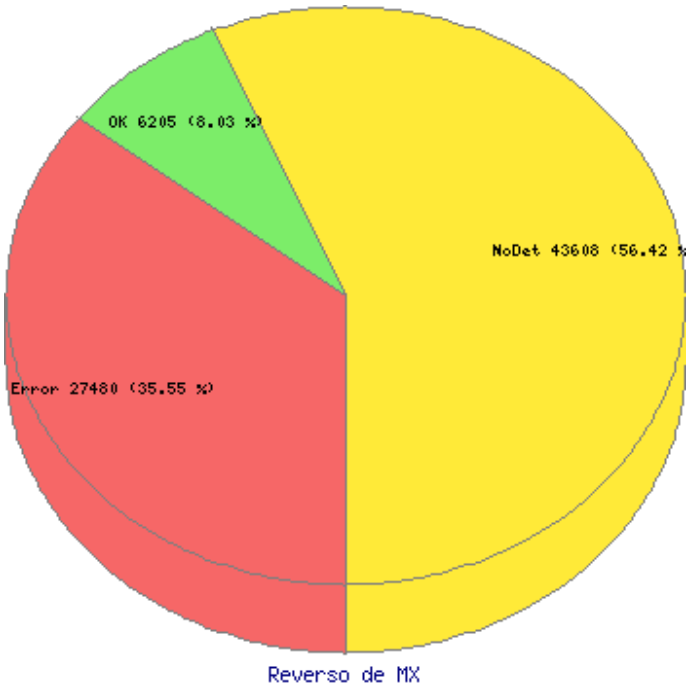

Figura 8.23: Evaluación de Reverso del Registro MX

8.5. Código Fuente de la Herramienta

A continuación, se anexa el código fuente de la herramienta. Este no incluye las bibliotecas de funciones ni de configuración.

```
#!/usr/bin/perl

use Net::DNS;
use POSIX;
use DBI;
use DNS::ZoneParse;
use IO::Select;


use LibInspector;


use strict;

print "Iniciando...\n";

my $db = LibInspector::conectaBaseDatos();

print "Conectado a la base de datos\n";

LibInspector::creaTablasInspector($db);

print "Creadas las tablas\n";

LibInspector::insertaCaracteristicas($db);

print "Insertadas las caracteristicas\n";

LibInspector::insertaDominioServidor($db);

print "Insertados dominios y servidores\n";

# Comenzamos con las evaluaciones

open (LOG, ">log-".strftime( "%Y%m%d%H%M%S", localtime(time)).".txt")
    || die "No logre abrir log $!\n";
```

```
my @dominios = LibInspector::dominiosEvaluar($db);
my @servidoresMalos;
print "\n----\nEvaluando Completitud de Servidores, para: ". $#dominios
    ." dominios\n";
foreach my $dom (@dominios){
  # inicializo variables de la evaluacion y las glosa
  my @evaluacion = LibInspector::inicializaEvaluacion();
  my @glosa = LibInspector::inicializaGlosa();

  # listado de servidores para el dominio
  print "analizando dominio : $dom\n";
  my @servidores = LibInspector::servidoresDominio($db, $dom);
  print "La base de datos dice: ".join("**", @servidores). "\n";
  $glosa[1] = "<p>En la zona los servidores de nombres son: ".
              join(", ", @servidores);

  # CARACTERISTICA 4
  # Numero de servidores
  my $nroservidores = $#servidores + 1;
  if ($nroservidores >= 2 && $nroservidores <= 7 ){
    $evaluacion[4] = 1;
    $glosa[4] = "<p>El nombre de dominio tiene: $nroservidores
       servidores de nombres</p>";
  } else {
    $evaluacion[4] = 0;
    $glosa[4] = "<p>El nombre de dominio tiene: $nroservidores
       servidores de nombres</p>";
  }

  # para almacenar las versiones del software de DNS
  my @version;
  my @serial;
  my $soamname = undef;
  foreach my $ser (@servidores){
    # consultamos a cada servidor por el NS
    print "preguntando a un servidor: $ser\n";
    my @servidoresNS;
    if (!grep(/^$ser$/, @servidoresMalos)){
      my $res = Net::DNS::Resolver->new;
      $res->nameservers($ser);

      # CARACTERISTICA 3
      # Vemos la IP del servidor
      my @ips = $res->nameservers;
```

```
if ($#ips > -1){
  $glosa[3] .= "<p>El servidor: $ser, tiene la(s) siguientes
     direcciones IP: ".
              join(", ", @ips)."</p>";
  # Buscamos el reverso de las IP

  foreach my $ipRev (@ips){
    my $nombreReverso = reversoIP($ipRev);
    if ($nombreReverso ne ""){
      if ($evaluacion[3] != 0) {
        if (lc($nombreReverso) eq lc($ser)){
          $evaluacion[3] = 1;
        } else {
          # Encontramos el reverso, pero el nombre es distinto al
              servidor
          $evaluacion[3] = 0;
        }
      }
      $glosa[3] .= "<p>Para la ódireccin IP: $ipRev se óencontr el
          reverso: $nombreReverso</p>";
    }  else {
      $evaluacion[3] = 0;
      $glosa[3] .= "<p>Para la ódireccin IP: $ipRev NO se óencontr
          el reverso</p>";
    }
  }

  # insertar las IP en la base de datos
  LibInspector::insertaServidorIp($db, $ser, \@ips);
} else {
  $evaluacion[3] = 0;
  $glosa[3] .= "<p>Error, imposible encontrar reverso para $ser</
     p>";
  # insertar el servidor en los servidores malos
  push(@servidoresMalos, lc($ser));
}
$res->udp_timeout(3);
my $answer = $res->query($dom, 'NS');

if ($answer) {
  # RESPONDE :-)
  foreach my $rr ($answer->answer) {
    if ($rr->type eq "NS"){
      print "Servidor NS: ", $rr->nsdname, "\n";
```

```
    push (@servidoresNS, lc($rr->nsdname).".");
  }
}
# CARACTERISTICA 2
# Servidor responde, se debe marcar como OK, la evaluacion
# de la caracteristica 2
$evaluacion[2] = 1 if ($evaluacion[2] != 0);
$glosa[2] .= "<p>El servidor: $ser esta respondiendo</p>";

# CARACTERISTICA 5
# obtener los headers de la respuesta y analizar si la
   respuesta es
# autoritativa

my $header = $answer->header;
if ($header->aa){
  $evaluacion[5] = 1 if ($evaluacion[5] != 0);
  $glosa[5] .= "<p>El servidor $ser responde con autoridad</p>"
     ;
} else {
  $evaluacion[5] = 0;
  $glosa[5] .= "<p>El servidor $ser responde SIN autoridad</p>"
     ;
}

# CARACTERISTICA 6
# obtener los headers de la respuesta y analizar si la
   respuesta es
# recursiva
if (!$header->ra){
  $evaluacion[6] = 1 if ($evaluacion[6] != 0);
  $glosa[6] .= "<p>El servidor $ser responde sin recursividad</
     p>";
} else {
  $evaluacion[6] = 0;
  $glosa[6] .= "<p>El servidor $ser responde con recursividad</
     p>";
}

# CARACTERISTICA 7
# NS como CNAME
my $rescn = Net::DNS::Resolver->new;
my $answercn = $res->query($ser);
if ($answercn){
```

```
    foreach my $rr ($answercn->answer) {
      if ($rr->type eq "CNAME"){
        $evaluacion[7] = 0;
        $glosa[7] .= "<p>Error, servidor $ser, definido como
           CNAME:<br><pre>".
        $rr->string ."</pre></p>";
      }
    }
  } else {
    $evaluacion[7] = 0;
    $glosa[7] .= "<p>Error, no pude determinar si el servidor
       $ser, áest definido ".
                "como CNAME: ". $rescn->errorstring. "</p>";
  }

  if ($evaluacion[7] != 0){
    $evaluacion[7] = 1;
    $glosa[7] .= "<p>El servidor $ser, no áest definido como CNAME
       </p>";
  }


  # CARACTERISTICA 9
  # versiones de DNS

  my $packet = Net::DNS::Packet->new("version.bind", "TXT", "CH")
     ;
  my $answerver = $res->send($packet);
  if ($answerver){
    # Respondio, ahora hay que ver si la respuesta es vacia
    my @rver = $answerver->answer;
    if ($#rver > -1){
      # Procesar la respuesta
      foreach my $rr ($answerver->answer) {
        $glosa[9] .= "<p>Version de Software encontrada en $ser:
           ". $rr->rdatastr.
        "</p>";
        push (@version, $rr->rdatastr);
      }
    }
    else{
      # intentar con otra consulta
      $packet = Net::DNS::Packet->new("id.server", "TXT", "CH");
      $answerver = $res->send($packet);
```

```
      if ($answerver){
        @rver = $answerver->answer;
        if ($#rver > -1){
          foreach my $rr ($answerver->answer) {
            $glosa[9] .= "<p>óVersin de Software encontrada en
                $ser: ".
            $rr->rdatastr. "</p>";
            push (@version, $rr->rdatastr);
          }
        }
        else{
          $glosa[9] .= "<p>Error, no determine óversin de Software
              DNS para $ser</p>";
          $evaluacion[9] = 0;
        }
      }
      else{
        $glosa[9] .= "<p>Error, no determine óversin de Software
            DNS para $ser</p>";
        $evaluacion[9] = 0;
      }

    }
  } else {
    $glosa[9] .= "<p>Error, no determine óversin de Software DNS
        para $ser</p>";
    $evaluacion[9] = 0;
  } # fin versiones NS


  # CARACTERISTICA 18
  # WWW como CNAME
  my $reswww = Net::DNS::Resolver->new;
  $reswww->nameserver($ser);
  $reswww->udp_timeout(3);
  print "EV18, preguntando por www.$dom, A\n";
  my $answerwww = $reswww->query("www.".$dom, "A");
  if ($answerwww){
    foreach my $rr ($answerwww->answer) {
      if ($rr->type eq "CNAME"){
        $evaluacion[18] = 0;
        $glosa[18] .= "<p>Error, el registro WWW para $dom en
            $ser, áest definido como CNAME:<br><pre>".
        $rr->string ."</pre></p>";
```

```
    }
  }
} else {
  $evaluacion[18] = 0;
  $glosa[18] .= "<p>Error, no pude determinar si el registro
    WWW esta definido en $ser ".
            "como CNAME: ". $rescn->errorstring. "</p>";
}

if ($evaluacion[18] != 0){
  $evaluacion[18] = 1;
  $glosa[18] .= "<p>El registro WWW no áest definido como CNAME
    en $ser</p>";
}

# CARACTERISTICA 19 y 20
# MX como CNAME
my $resmx = Net::DNS::Resolver->new;
$resmx->nameserver($ser);
$resmx->udp_timeout(3);
print "EV19, preguntando por MX de $dom\n";
my $answermx = $resmx->query($dom, "MX");
if ($answermx){
  my $contEV22 = 0;
  $evaluacion[19] = 1 if ($evaluacion[19] != 0);
  my $rrmx = ($answermx->answer)[0];
  $glosa[19] .= "<p>Existe registro MX para el nombre de
    dominio $dom:<br><pre>". $rrmx->string ."</pre></p>";
  foreach my $rr ($answermx->answer) {
    # Incrementamos el contador de servidores MX del dominio
    $contEV22++ if ($rr->type eq "MX");
    if ($rr->type eq "CNAME"){
      $evaluacion[20] = 0;
      $glosa[20] .= "<p>Error, el registro MX para $dom en $ser
        , áest definido como CNAME:<br><pre>".
      $rr->string ."</pre></p>";
    }
  }

  # CARACTERISTICA 21
  # Si respondio, se debe averiguar si el nombre del MX
    corresponde a un nombre de host
  my $servidorMX = $rrmx->string;
  $servidorMX =~ /([^ ]+)$/;
```

```
my $nombreServidorMX = $1;
if (LibInspector::esNombreServidor($nombreServidorMX)){
  $evaluacion[21] = 1 if ($evaluacion[21] != 0);
  $glosa[21] .= "<p>El nombre del servidor MX:<i>".
     $nombreServidorMX."</i> es un nombre de Host</p>";
} else {
  $evaluacion[21] = 0 if ($evaluacion[21] != 1);
  $glosa[21] .= "<p>El nombre del servidor MX:<i>".
     $nombreServidorMX."</i> no es un nombre de Host</p>";
}

# CARACTERISTICA 22
if ($contEV22 > 1){
  $evaluacion[22] = 1 if ($evaluacion[22] != 0);
  $glosa[22] .= "<p>El únmero de servidores MX es: ".$contEV22
     ." en el servidor: $ser</p>";
} else{
  $evaluacion[22] = 0 if ($evaluacion[22] != 1);
  $glosa[22] .= "<p>El únmero de servidores MX es: ".$contEV22
     ." en el servidor: $ser</p>";
}

# CARACTERISTICA 23
# El reverso del MX

my $revMX = Net::DNS::Resolver->new;
$revMX->nameservers($nombreServidorMX);
my @ipsMX = $revMX->nameservers;
if ($#ipsMX > -1){
  $glosa[23] .= "<p>El servidor MX $nombreServidorMX, tiene
     las siguientes direcciones IP: "
                .join(", ", @ipsMX)."</p>";
  # Buscamos el reverso de las IP
  foreach my $ipRev (@ipsMX){
    my $nombreReverso = reversoIP($ipRev);
    if ($nombreReverso ne ""){
      if ($evaluacion[23] != 0) {
        if (lc($nombreReverso) eq lc($nombreServidorMX)){
          $evaluacion[23] = 1;
        } else {
          # Encontramos el reverso, pero el nombre es
             distinto al
          # servidor
          $evaluacion[23] = 0;
```

```
            }
          }
          $glosa[23] .= "<p>Para la ódireccin IP: $ipRev se
             óencontr el reverso: $nombreReverso</p>";
        } else {
          $evaluacion[23] = 0;
          $glosa[23] .= "<p>Para la ódireccin IP: $ipRev NO se
             óencontr el reverso</p>";
        }
      }
    } else {
      $evaluacion[23] = 0;
      $glosa[23] .= "<p>Error, imposible encontrar reverso para
         $ser</p>";
    }

  } else {
    $evaluacion[19] = 0;
    $glosa[19] .= "<p>Error, no pude encontrar registro MX para
       $dom en $ser</p>";

    $evaluacion[20] = 0;
    $glosa[20] .= "<p>Error, no pude determinar si el registro MX
        esta definido en $ser ".
                "como CNAME: ". $rescn->errorstring. "</p>";

    $evaluacion[21] = 0;
    $glosa[21] .= "<p>Error, no pude encontrar registro MX para
       $dom en $ser</p>";

    $evaluacion[22] = 0 if ($evaluacion[22] != 1);
    $glosa[22] .= "<p>Error, no pude encontrar registro MX para
       $dom en $ser</p>";

    $evaluacion[23] = 0 if ($evaluacion[23] != 1);
    $glosa[23] .= "<p>Error, no pude encontrar registro MX para
       $dom en $ser</p>";
  }

  if ($evaluacion[20] != 0){
    $evaluacion[20] = 1;
    $glosa[20] .= "<p>El registro MX no áest definido como CNAME
       en $ser</p>";
  }
```

```
# CARACTERISTICA 10 - 17
# Analisis del SOA

my $answersoa = $res->query($dom, "SOA");
if ($answersoa){
  my $rr = ($answersoa->answer)[0];
  push(@serial, $rr->serial);
  $glosa[10] .= "<p>El servidor $ser dice que serial es: ". $rr
     ->serial ."</p>";


  # Caracteristica C11
  $glosa[11] .= "<p>El servidor $ser, dice que el nombre del
     servidor primario ".
           "es: ". $rr->mname ."</p>";
  my $auxser = $rr->mname.".";
  # vemos si el mname esta entre los servidores del dominio
  if (grep(/^$auxser$/, @servidores)){
    $evaluacion[11] = 1;
  }
  else{
    $evaluacion[11] = 0 if ($evaluacion[11] != 1);
  }

  # Caracteristica 12
  if ($evaluacion[12] != 1){
    if (length($rr->serial) == 10){
      my $serialyear = substr($rr->serial, 0, 4);
      my $serialmonth = substr($rr->serial, 4, 2);
      my $serialday = substr($rr->serial, 6, 2);
      my ($ye) = strftime( "%Y", localtime(time));
      if (($serialyear <= $ye) && ($serialyear > $ye-10) &&
          ($serialmonth <= 12) && ($serialmonth > 0) &&
          ($serialday <=31) && ($serialday > 0)) {
          $glosa[12] .= "<p>El numero serial ".$rr->serial."
             parece cumplir con el ".
                       "formato AAAAMMDDnn, en el servidor
                          $ser</p>";
          $evaluacion[12] = 1;
      }
    } else {
```

```
        $glosa[12] .= "<p>Error, el numero serial: ".$rr->serial.
            " parece no tener el formato de ".
                    "AAAAMMDDnn, en el servidor $ser</p>";
        $evaluacion[12] = 0;
      }
    }

    # Caracteristica 13
    if ($evaluacion[13] != 1){
      if (($rr->retry < $rr->refresh) && ($rr->refresh < $rr->
          minimum) &&
           ($rr->minimum < $rr->expire)){
        $glosa[13] .= "<p>Los tiempos del registro SOA cumplen
            con reintento &lt; refresco &lt; ttl ".
                    "&lt; expiracion (". $rr->retry ." &lt; ".
                        $rr->refresh ." &lt; ". $rr->minimum .
                    " &lt; ".
                    $rr->expire ."), en el servidor $ser</p>";
        $evaluacion[13] = 1;

      } else {
        $glosa[13] .= "<p>Error, los tiempos del SOA no cumplen
            con  reintento &lt; refresco &lt; ttl ".
                    "&lt; expiracion en servidor $ser</p>";
        $evaluacion[13] = 0;
      }
    }

    # Caracteristica 14
    if ($evaluacion[14] != 1){
      if (($rr->refresh >= 3600) && ($rr->refresh <= 7200)){
        $glosa[14] .= "<p>El tiempo de refresco: ".$rr->refresh .
            " áest dentro de los áparmetros ".
                    "correctos (3600 &lt; refresco &gt; 7200)</p>
                        ";
        $evaluacion[14] = 1;
      } else {
        $glosa[14] .= "Error, el tiempo de refresco: ".$rr->
            refresh ." no áest dentro de los áparmetros ".
                    "correctos (3600 &lt; refresco &gt; 7200),
                        en el servidor $ser</p>";
        $evaluacion[14] = 0;
      }
    }
```

```
# Caracteristica 15
if ($evaluacion[15] != 1){
  if (($rr->retry >= 120) && ($rr->retry <= 7200)){
    $glosa[15] .= "<p>El tiempo de reintento: ".$rr->retry ."
        áest dentro de los áparmetros ".
                "correctos (120 &lt; reintento &gt; 7200)</p>
                  ";
    $evaluacion[15] = 1;
  } else {
    $glosa[15] .= "Error, el tiempo de reintento: ".$rr->
       retry ." no áest dentro de los áparmetros ".
                 "correctos (120 &lt; reintento &gt; 7200),
                    en el servidor $ser</p>";
    $evaluacion[15] = 0;
  }
}

# Caracteristica 16
if ($evaluacion[16] != 1){
  if (($rr->expire >= 1209600) && ($rr->expire <= 2419200)){
    $glosa[16] .= "<p>El tiempo de expiracion: ".$rr->expire
       ." áest dentro de los áparmetros ".
                "correctos (1209600 &lt; expiracion &gt;
                   2419200)</p>";
    $evaluacion[16] = 1;
  } else {
    $glosa[16] .= "Error, el tiempo de expiracion: ".$rr->
       expire ." no áest dentro de los áparmetros ".
                 "correctos (1209600 &lt; expiracion &gt;
                    2419200), en el servidor $ser</p>";
    $evaluacion[16] = 0;
  }
}

# Caracteristica 17
if ($evaluacion[17] != 1){
  if (($rr->minimum >= 86400) && ($rr->minimum <= 432000)){
    $glosa[17] .= "<p>El tiempo de vida de la óinformacin: ".
       $rr->minimum ." áest dentro de los ".
                "áparmetros correctos (86400 &lt; ttl &gt;
                   432000)</p>";
    $evaluacion[17] = 1;
  } else {
```

```
            $glosa[17] .= "Error, el tiempo de vida de la óinformacin:
                ".$rr->minimum ." no áest dentro ".
                        "de los áparmetros correctos (86400 &lt;
                           minimum &gt; 43200), en el ".
                        "servidor $ser</p>";
            $evaluacion[17] = 0;
          }
        }


      } else {
        $evaluacion[10] = 0;
        $glosa[10] .= "<p>Error, no pude determinar únmero serial del
           SOA para $dom en ".
                    "$ser</p>";

        $evaluacion[11] = 0 if ($evaluacion[11] != 1);
        $glosa[11] .= "<p>Error, no pude determinar el nombre de
           servidor en SOA, en: ".
                    "$ser</p>";

        $evaluacion[12] = 0 if ($evaluacion[12] != 1);
        $glosa[12] .= "<p>Error, no pude determinar el numero serial,
            en servidor: ".
                    "$ser</p>";

        $evaluacion[13] = 0 if ($evaluacion[13] != 1);
        $glosa[13] .= "<p>Error, no se pueden determinar los tiempos
           del SOA en $ser</p>";

        $evaluacion[14] = 0 if ($evaluacion[14] != 1);
        $glosa[14] .= "<p>Error, no se pueden determinar el tiempo de
            refresco del SOA en $ser</p>";

        $evaluacion[15] = 0 if ($evaluacion[15] != 1);
        $glosa[15] .= "<p>Error, no se pueden determinar el tiempo de
            reintento del SOA en $ser</p>";

        $evaluacion[16] = 0 if ($evaluacion[16] != 1);
        $glosa[16] .= "<p>Error, no se pueden determinar el tiempo de
            expiracion del SOA en $ser</p>";

        $evaluacion[17] = 0 if ($evaluacion[17] != 1);
        $glosa[17] .= "<p>Error, no se pueden determinar el tiempo de
```

```
        expiracion del SOA en $ser</p>";
    }


}
else {
    $glosa[1] .= "<br>Error al consultar $ser: ". $res->errorstring
        ."<br>";
    print "query failed: ", $res->errorstring, "\n";
    push(@servidoresMalos, lc($ser)) if ($res->errorstring eq "no
       nameservers");

    # CARACTERISTICA 2
    # Servidor no responde, se debe marcar como mala la evaluacion
    # de la caracteristica 2
    $evaluacion[2] = 0;
    $glosa[2] .= "<p>Error, el servidor: $ser, NO áest respondiendo:
        ". $res->errorstring ."</p>";

    # CARACTERISTICA 5
    $evaluacion[5] = 0;
    $glosa[5] .= "<p>Error, el servidor: $ser responde: ". $res->
       errorstring."</p>";

    # CARACTERISTICA 6
    $evaluacion[6] = 0;
    $glosa[6] .= "<p>El servidor $ser responde: ". $res->
       errorstring.", imposible analizar recursividad</p>";

    # CARACTERISTICA 7
    $evaluacion[7] = 0;
    $glosa[7] .= "<p>Error, no pude determinar si el servidor $ser,
        áest definido ".
            "como CNAME: ". $res->errorstring ."</p>";

    # CARACTERISTICA 9
    $evaluacion[9] = 0;
    $glosa[9] .= "<p>El servidor $ser NO áest respondiendo</p>";

    # CARACTERISTICA 18
    $evaluacion[18] = 0;
    $glosa[18] .= "<p>Error, no pude determinar si el registro WWW
       áest definido ".
            "como CNAME: ". $res->errorstring ." en $ser</p>";
```

```
# CARACTERISTICA 19
$evaluacion[19] = 0;
$glosa[19] .= "<p>Error, no pude determinar si el registro MX
   áest definido".
          ": ". $res->errorstring ." en $ser</p>";

# CARACTERISTICA 20
$evaluacion[20] = 0;
$glosa[20] .= "<p>Error, no pude determinar si el registro MX
   áest definido ".
          "como CNAME: ". $res->errorstring ." en $ser</p>";

# CARACTERISTICA 21
$evaluacion[21] = 0 if ($evaluacion[21] != 1);
$glosa[21] .= "<p>Error, no pude determinar si el registro MX
   áest definido".
          ": ". $res->errorstring ." en $ser</p>";

# CARACTERISTICA 22
$evaluacion[22] = 0 if ($evaluacion[22] != 1);
$glosa[22] .= "<p>Error, no pude determinar si el registro MX
   áest definido".
          ": ". $res->errorstring ." en $ser</p>";

# CARACTERISTICA 23
$evaluacion[23] = 0 if ($evaluacion[23] != 1);
$glosa[23] .= "<p>Error, no pude determinar si el registro MX
   áest definido".
          ": ". $res->errorstring ." en $ser</p>";

# CARACTERISTICA 10
$evaluacion[10] = 0;
$glosa[10] .= "<p>Error, no pude determinar únmero serial del
   SOA para $dom en ".
          "$ser</p>";

# CARACTERISTICA 11
$evaluacion[11] = 0 if ($evaluacion[11] != 1);
$glosa[11] .= "<p>Error, no pude determinar el nombre de
   servidor en SOA, en: ".
          "$ser</p>";

# CARACTERISTICA 12
```

```
        $evaluacion[12] = 0 if ($evaluacion[12] != 1);
        $glosa[12] .= "<p>Error, no pude determinar el numero serial,
           en servidor:  ".
                    "$ser</p>";

        # CARACTERISTICA 13
        $evaluacion[13] = 0 if ($evaluacion[13] != 1);
        $glosa[13] .= "<p>Error, imposible analizar los tiempos del SOA
           , $ser no áest respondiendo</p> ";

        # CARACTERISTICA 14
        $evaluacion[14] = 0 if ($evaluacion[14] != 1);
        $glosa[14] .= "<p>Error, imposible analizar los tiempos del SOA
           , $ser no áest respondiendo</p> ";

        # CARACTERISTICA 15
        $evaluacion[15] = 0 if ($evaluacion[15] != 1);
        $glosa[15] .= "<p>Error, imposible analizar los tiempos del SOA
           , $ser no áest respondiendo</p> ";

        # CARACTERISTICA 16
        $evaluacion[16] = 0 if ($evaluacion[16] != 1);
        $glosa[16] .= "<p>Error, imposible analizar los tiempos del SOA
           , $ser no áest respondiendo</p> ";

        # CARACTERISTICA 17
        $evaluacion[17] = 0 if ($evaluacion[17] != 1);
        $glosa[17] .= "<p>Error, imposible analizar los tiempos del SOA
           , $ser no áest respondiendo</p> ";
    }
}
else{
    # Si el servidor a consultar esta entre los servidores malos
    $glosa[1] .= "<br>Error al consultar $ser: no nameservers<br>";
    print "pregunte a servidor identificado como malo: $ser\n";

    # CARACTERISTICA 2
    $evaluacion[2] = 0;
    $glosa[2] .= "El servidor: $ser NO áest respondiendo</p>";

    # CARACTERISTICA 3
    $evaluacion[3] = 0;
    $glosa[3] .= "<p>Error, imposible determinar reverso para el
       servidor: $ser</p>";
```

```
# CARACTERISTICA 5
$evaluacion[5] = 0;
$glosa[5] .= "<p>Error, el servidor: $ser no áest respondiendo</p>
    ";

# CARACTERISTICA 6
$evaluacion[6] = 0;
$glosa[6] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 7
$evaluacion[7] = 0;
$glosa[7] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 9
$evaluacion[9] = 0;
$glosa[9] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 18
$evaluacion[18] = 0;
$glosa[18] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 19
$evaluacion[19] = 0;
$glosa[19] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 20
$evaluacion[20] = 0;
$glosa[20] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 21
$evaluacion[21] = 0 if ($evaluacion[21] != 1);
$glosa[21] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 22
$evaluacion[22] = 0 if ($evaluacion[22] != 1);
$glosa[22] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 23
$evaluacion[23] = 0 if ($evaluacion[23] != 1);
$glosa[23] .= "<p>El servidor $ser NO áest respondiendo</p>";

# CARACTERISTICA 10
$evaluacion[10] = 0;
```

```
    $glosa[10] .= "<p>Error, no pude determinar únmero serial del SOA
       para $dom en ".
                "$ser</p>";

    # CARACTERISTICA 11
    $evaluacion[11] = 0 if ($evaluacion[11] != 1);
    $glosa[11] .= "<p>Error, no pude determinar el nombre de servidor
        en SOA, en: ".
                "$ser</p>";

    # CARACTERISTICA 12
    $evaluacion[12] = 0 if ($evaluacion[12] != 1);
    $glosa[12] .= "<p>Error, no pude determinar el numero serial, en
       servidor:  ".
                "$ser</p>";

    # CARACTERISTICA 13
    $evaluacion[13] = 0 if ($evaluacion[13] != 1);;
    $glosa[13] .= "<p>Error, imposible analizar los tiempos del SOA,
       $ser no áest respondiendo</p> ";

    # CARACTERISTICA 14
    $evaluacion[14] = 0 if ($evaluacion[14] != 1);
    $glosa[14] .= "<p>Error, imposible analizar los tiempos del SOA,
       $ser no áest respondiendo</p> ";

    # CARACTERISTICA 15
    $evaluacion[15] = 0 if ($evaluacion[15] != 1);
    $glosa[15] .= "<p>Error, imposible analizar los tiempos del SOA,
       $ser no áest respondiendo</p> ";

    # CARACTERISTICA 16
    $evaluacion[16] = 0 if ($evaluacion[16] != 1);
    $glosa[16] .= "<p>Error, imposible analizar los tiempos del SOA,
       $ser no áest respondiendo</p> ";

    # CARACTERISTICA 17
    $evaluacion[17] = 0 if ($evaluacion[17] != 1);
    $glosa[17] .= "<p>Error, imposible analizar los tiempos del SOA,
       $ser no áest respondiendo</p> ";
}
print "numero de servidores: ". $#servidoresNS ." y deberian ser: "
   . $#servidores."\n";
print "servidores encontrados: ".join("||", @servidoresNS)."\n";
```

```
  # ANALISIS CARACTERISTICA 1
  # Si el numero de servidores son distintos o el grupo de servidores
      son
  # diferentes la evaluacion esta mala
  if (($#servidores != $#servidoresNS) ||
      LibInspector::arreglosDistintos(\@servidores, \@servidoresNS)){
    $glosa[1] .= "<p>Error, $ser dice que los registros NS son: ".
          join(", ", @servidoresNS)."<br>";
    print "Distinto numero de servidores, en $ser: deberian ser ".
    join("||", @servidores)." y son: ". join("||", @servidoresNS)."\
       n";
    join("||", @servidores)." y son: ". join("||", @servidoresNS)."\
       n";
    $evaluacion[1] = 0;
  }
  else{
    $glosa[1] .= "<br>Servidor $ser dice que los registros NS son: ".
          join(", ", @servidoresNS);
    print "Test OK\n";
    $evaluacion[1] = 1 if ($evaluacion[1] != 0);
  }
  $glosa[1] .="</p>";
}
# Analizar las caracteristicas que dependian de la respuesta de todos
    los servidores

# C8 Clases de direcciones IP
my @ipsDom = LibInspector::direccionesIPdominio($db, $dom);
@ipsDom = sort(@ipsDom);
my $auxNew = "";
my $auxOld = "";
my $okc8 = 0;
foreach my $ipd (@ipsDom){
  print "EV8: Revisando ip $ipd\n";
  my ($pre1, $pre2, $pre3, $pre4) = split(/\./, $ipd);
  $auxNew = $pre1 ."."." $pre2 ."."." $pre3;
  $okc8 = 1 if (($auxNew ne $auxOld) && ($okc8 != 1) && ($auxOld ne "
     "));
  $auxOld = $auxNew;
}
if ($#ipsDom > 0){
  if ($okc8){
    $evaluacion[8] = 1;
```

```
    $glosa[8] .= "<p>Las direcciones IP de los servidores de nombres
       son: <i>".
                join(", ", @ipsDom)."</i> las que parecen estar en
                   redes diferentes.</p>";
  } else {
    $evaluacion[8] = 0;
    $glosa[8] .= "<p>Las direcciones IP de los servidores de nombres
       son: <i>".
                join(", ", @ipsDom)."</i> las que parecen estar en
                   redes iguales.</p>";
  }
} else {
  $evaluacion[8] = 0;
  $glosa[8] .= "<p>Existe 1 ódireccin IP para el servidor de nombres,
     la que es: <i>".
              join(", ", @ipsDom)."</i>.</p>";
}


# C9 Version del SW DNS
if ($#version > -1){
  if ($#version == 0){
    # Si solo hay un servidor consideramos OK
    $evaluacion[9] = 1;
  }  else {
    for (my $i=0; $i < $#version; $i++){
      # con 1 que sea distinto damos OK el test
      if ($version[$i] ne $version[$i+1]){
        $evaluacion[9] = 1;
      }
    }
  }
}

# Analizar C10, arreglo de seriales
if ($#serial > -1){
  if ($#serial == 0 && $evaluacion[10] != 0){
    # Solo un serial, test OK
    $evaluacion[10] = 1;
  }  else {
    if ($evaluacion[10] != 0){
      #suponemos que esta OK y revisamos con los demas
      $evaluacion[10] = 1;
      for (my $i=0; $i < $#serial; $i++){
```

```
            # con 1 que sea distinto damos Malo el test
            if ($serial[$i] ne $serial[$i+1]){
              $evaluacion[10] = 0;
            }
          }
        }
      }
    }


    # Aca se debe insertar en la base de datos la glosa el resultado de
       la
    # evaluacion

    for (my $i=1; $i <= $#evaluacion; $i++){
      LibInspector::insertaEvaluacion($db, $dom, $i, $evaluacion[$i],
         $glosa[$i]);
    }

    print "\n----\n\n";
}
close(LOG);


$db->disconnect;


##############################################
# Obtiene el reverso de una direccion IP

sub reversoIP{
  my ($ip) = @_;
  my $result = "";
  my $timeout = 5;

  my $res = new Net::DNS::Resolver;
  my $bgsock = $res->bgsend( $ip );
  my $sel = new IO::Select($bgsock);
  my @ready = $sel->can_read($timeout);
  if (@ready) {
    foreach my $sock (@ready) {
      if ($sock == $bgsock) {
        my $packet = $res->bgread($bgsock);
        if ($packet) {
```

```
            foreach my $rr ($packet->answer) {
              $result = $rr->rdatastr;
            }
          }
          else {
            $result = $ip;
          }
          $bgsock = undef;
        }
        $sel->remove($sock);
        $sock = undef;
      }
    }
    return $result;

}
```

Bibliografía

[1] Paul Albitz y Cricket Liu. **DNS and BIND, 4th Edition**.

[2] David Barr. **Common DNS Operational and Configuration Errors**. Disponible a través de `http://www.ietf.org/rfc/rfc1912.txt?number=1912`

[3] P. Mockapetris **Domain Names - Concepts and Facilities**. Disponible a través de `http://www.ietf.org/rfc/rfc1034.txt?number=1034`

[4] P. Mockapetris **Domain Names - Implementation and Specification**. Disponible a través de `http://www.ietf.org/rfc/rfc1035.txt?number=1035`

[5] Jonathan B. Postel **Simple MAIL Transfer Protocol**. Disponible a través de `http://www.ietf.org/rfc/rfc821.txt?number=821`

[6] David H. Crocker **Standard for ARPA Internet Text Messages**. Disponible a través de `http://www.ietf.org/rfc/rfc822.txt?number=822`

[7] Craig Partridge **Mail Routing and the Domain System**. Disponible a través de `http://www.ietf.org/rfc/rfc974.txt?number=974`